Multiclass Feature Selection with Kernel Gram-matrix-based criteria

Mathieu Ramona, *Member, IEEE,*
Gaël Richard, *Senior Member, IEEE,*
and Bertrand David, *Member, IEEE*

***Abstract*—Feature selection has been an important issue during the last decades to determine the most relevant features according to a given classification problem. Numerous methods emerged that take into account Support Vector Machines in the selection process. Such approaches are powerful but often complex and costly. In this paper, we propose new feature selection methods based on two criteria designed for the optimization of SVM: Kernel Target Alignment and Kernel Class Separability. We demonstrate how these two measures, when fully expressed, can build efficient and simple methods, easily applicable to multiclass problems, and iteratively computable with minimal memory requirements. An extensive experimental study is conducted both on artificial and real-world data sets to compare the proposed methods to state of the art feature selection algorithms. The results demonstrate the relevance of the proposed methods both in terms of performance and computational cost.**

***Index Terms*—Feature Selection, Variable Selection, Support Vector Machines, Kernel Target Alignment, Kernel Class Separability, Audio classification**

I. Introduction

In the context of supervised pattern recognition, the gathering of large data sets has become a common process with the availability of more sensors and the increase of computational resources. But the accumulation of data is not necessarily profitable for pattern recognition systems, which generally face the so-called *curse of dimensionality* (explained in [1] by the fact that a high-dimensional space, populated by a finite set, is nearly empty). Sparseness of the training set results in the classifier's overfitting and thus penalizes generalization. Moreover, large collections of features generally contain highly correlated descriptors derived from the same sources, or irrelevant ones, feeding the learning process with unreliable information.

Feature selection aims at determining the most relevant features according to a given problem. The dimension reduction and the removal of irrelevant features are meant to enhance generalization performances but also allow some insights on the problem through the interpretation of the most relevant features. This also yields an important cost reduction both in storage need and computational speed.

According to John et al. [2] and Guyon and Elisseeff [3], feature selection methods divide between *filters*, built as preprocessing steps of the classification and thus independent of the classifier, and *wrappers* that use the classifier as a black box to operate the feature selection. However, even *filter* selection is related to the classifier, as the selection criterion is always based on an assumption on the classification process.

Mathieu Ramona is with the Sound Analysis/Synthesis Team, IRCAM/CNRS-STMS, Paris 75004, France. (e-mail: mathieu.ramona@ensta.org). Part of this work was done when M. Ramona was at Institut Mines-Telecom, Telecom ParisTech, CNRS LTCI, France.

Gaël Richard and Bertrand David are with Institut Mines-Telecom, Telecom ParisTech, CNRS LTCI, France.

Linear Discriminant aims at determining an optimal hyperplane separating both classes' examples, but the choice of the optimality criterion implies underlying assumptions. *Support Vector Machines* (SVM) lie on the distance between the separating hyperplane and the closest examples, the so-called *margin*. The problem, widely explored [4] [5], is solved through quadratic programming. The *kernel trick* further introduces non-linearity by substituting a kernel function $k(\boldsymbol{x}, \boldsymbol{y}) = \langle \Phi(\boldsymbol{x}), \Phi(\boldsymbol{y}) \rangle$ to inner products (where Φ can be implicit), under some restrictions over the choice of k. The target space of Φ is generally called the *feature space*, and has a much higher dimension (possibly infinite) than the original *input space*. This transformation widens the range and complexity of possible decision surfaces in the input space.

Several existing methods address the problem of taking the SVM underlying process into account in the feature selection step, among which the radius-margin bound [6], which shows very good results in practice. Nevertheless, they often involve multiple SVM trainings and even other optimization processes as part of the feature selection process, and are thus computationally expensive. Moreover, some are not designed to scale up to very large data sets. We propose here three new feature selection methods based on the Kernel Target Alignment and Kernel Class Separability criteria, that are evaluated iteratively from the sole Gram matrix values, and are thus simple, and very scalable in terms of memory. These two criteria have already been proposed to define feature selection algorithms, in particular by Neumann et al. [7] and Wang [8]. However, using simplifying assumptions, they only rely on a lower bound to the defined criteria. By contrast, the proposed methods in this paper are based on their original expression.

An extensive experimental study is conducted both on artificial and real-world data to compare the proposed methods to existing SVM-based feature selection techniques. The results demonstrate the relevance of the proposed criteria both in terms of performance and computational cost. We also show, both in theory and practice, that these methods are directly adaptable to problems involving more than two classes.

The rest of this paper is organized as follows. An overview of existing feature selection methods will be presented in Section II, prior to introducing the recent criteria used here and their application to feature selection in Sections III and IV. Numerical experiments on various data sets are detailed in Section V, and Section VI deals with computation issues, both in theory and practice. Some insights and perspectives are then given in Section VII.

II. Related works

A. Filter methods

The Fisher criterion, widely used on gene expression microarrays in bioinformatics [9], is one of the most classical filter approaches among feature selection methods. It is very similar to the Pearson correlation coefficients, which define a measure of correlation between the features and the labels, but also has the advantage of being symmetric in terms of classes:

$$r_F(d) = \frac{\left|\mu_{1,d} - \mu_{2,d}\right|^2}{\sigma_{1,d}^2 + \sigma_{2,d}^2}, \tag{1}$$

This criterion can be interpreted as the one-dimensional projection of the scatter-based Class Separability criterion $r_{CS}(d) = \frac{|\boldsymbol{\mu_1} - \boldsymbol{\mu_2}|^2}{|\boldsymbol{\sigma}_1^2 + \boldsymbol{\sigma}_2^2|}$. It is generally used to define a measure of relevance for each feature independently in order to rank them. This implies that interdependencies and redundancies between features are not considered.

Other classical filter methods in the literature are the Mutual Information [10] and the RELIEF algorithm [11], also based on statistical measures of correlation. More recent contributions include for example the Laplacian Score [12], based on the local structure of the data points, through a nearest neighbor graph. This algorithm lies on the assumption that close examples are prone to belong to the same class, and can therefore be used even for unsupervised selection.

B. SVM based feature selection

Whereas the previous filters are classifier-independent, many recent approaches are wrapper methods designed to take into account the specificities of the SVM classification process.

The principle of *Feature Selection concaVe* (FSV) [13] is closely related to SVM, and consists in jointly optimizing a separating hyperplane (of normal vector $\boldsymbol{w_h}$) and minimizing the so-called *zero-norm* (defined as the number of non-zero components) of $\boldsymbol{w_h}$. Being non-continuous, the zero-norm is approached by a concave function $||\boldsymbol{w_h}||_0 \approx \boldsymbol{1}^T(\boldsymbol{1} - e^{-\alpha \boldsymbol{w_h}})$. The authors further propose [14] to substitute the 2-norm $\frac{||\boldsymbol{w_h}||_2}{2}$ to the zero-norm, thus reaching the exact expression of an SVM optimization problem.

Using the components of the normal vector for feature ranking has been further investigated. SVM optimization consists in the determination of an optimal $\boldsymbol{w_h}$, expressed with the training examples $\boldsymbol{x_i}$, the corresponding class labels y_i, and the Lagrange multipliers α_i: $\boldsymbol{w_h} = \sum_i \alpha_i y_i \Phi(\boldsymbol{x_i})$.

A common problem lies in the fact that Φ is not explicit for most kernels. Several propositions bypass this issue by restricting their scope to linear kernels, among which the *Approximation of the zeRO-norm Minimization* (AROM) [15]. The authors substitute to a zero-norm minimization, the minimization of the value $\sum_{d=1}^{D} \ln|w_d|$ (where w_d are the components of the vector $\boldsymbol{w}_h$), and prove that both minima are nearly equal. The problem is solved by iteratively training an SVM on the component-weighted examples $\boldsymbol{x'} = \boldsymbol{w} \circ \boldsymbol{x}$ (where $\circ$ is the entry-wise product), while updating the scale vector components at each iteration with the normal vector: $\boldsymbol{w} \leftarrow \boldsymbol{w} \circ \boldsymbol{w_h}$. The problem can be expressed equivalently by introducing the *scaled* kernel $k_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}) = k(\boldsymbol{w} \circ \boldsymbol{x}, \boldsymbol{w} \circ \boldsymbol{y})$ and updating the *scale factors* w_d (i.e. the components of $\boldsymbol{w}$).

The *Recursive Feature Extraction* (RFE), proposed in [16], is based on the backward elimination of the features. The squared components of $\boldsymbol{w_h}$ are used as a criterion to evaluate the least relevant features to be iteratively discarded. The method is efficient and theoretically relevant, but at a much higher cost than AROM, not justified by its comparative performance [15].

Weston et al. also proposed [6] [17] a scale factors update strategy without expressing the vector $\boldsymbol{w_h}$. They use upper bounds to the leave-one-out estimate of the generalization error, among which the radius-margin bound $T = \frac{1}{n}\frac{R^2}{M^2}$, where M is the margin and R the radius of the smallest sphere containing the n training examples in the feature space. By derivating this bound with respect to the scale factors, they introduce a minimization problem to perform feature selection (referred to as R2W2).

Another interpretation of the scale factors is to consider the linear or RBF Gaussian kernels as combinations of *feature-wise* kernels. The field of Multiple Kernel Learning (MKL) [18] [19], originally restricted to the optimization of a conic linear combination of kernels, is one of the major trends in SVM research these last years, and its application for feature selection is straightforward. Rakotomamonjy et al. further introduced [20] a simple optimization method to solve MKL, that they evaluated on feature selection problems. Non-Monotonic feature selection [21] focuses on the development of an approximation to the combinatorial optimization problem to find the best binary combination of feature-wise kernels. One of the drawbacks of the scale factors is that the complexity is generally proportional to the number of features. Tan et al. propose [22] the Feature Generating Machine (FGM) method, adapted to very high dimensional datasets. Sparsity is incited in the MKL combination through the introduction of binary control variables instead of scale factors. The problem solving is done with a cutting plane algorithm. Varma and Babu [23] extend the MKL scheme to an even larger scope of combinations, including positive product of kernels, and also generalize the regularization on the kernel parameters. The so-called Generalized Multiple Kernel Learning (GMKL) method lies on the decomposition of the optimization problem into a nested two-step optimization loop. This method is inspired by the works of Chapelle et al. [17], where the inner-loop involves a standard SVM optimization procedure with constant combination coefficients. When restricted to positive product combinations with a L1-norm regularizer, the GMKL approach is equivalent to R2W2, although not based on the same optimization criterion.

Another scheme based on SVMs, consists in estimating the difference on posterior probabilities with and without each feature, in order to evaluate their ranking [24]. It is one of the few methods in the literature adapted to multiclass selection.

Finally, a few contributions take advantage of the Reproducing Kernel theory, i.e. of the kernel properties, without directly using Support Vector Machines. These can be seen as filter methods, since the classifier is not involved in the process, but they remain closely connected to the SVM theory. The FSKS algorithm [25] transposes the classical RELIEF method [11] and selects the features lying in the Kernel Space after a Kernel PCA projection. The BAHSIC method [26] consists in a backward elimination based on the Hilbert Space Information Criterion that estimates the dependence between features and labels in the Kernel Space.

Following the birth of the SVM theory, several measures were developed to tune the few parameters of the standard kernels. The radius-margin bound, introduced herebefore, is one of them, and was then extended to feature selection. The Kernel Target Alignment (KTA), introduced by Cristianini [27], and the Kernel Class Separability (KCS) are simple measures, solely based on the kernel Gram matrix, that prove very reliable for kernel tuning. However, very few attempts were made to use them for feature selection. Neumann et al. [7] and Wang [8] respectively propose algorithms based on the KTA and the KCS. However, in both cases, the practical constraints lead the authors to use only the numerator of the criteria, which is common between the two, and simpler to minimize. Moreover, in both cases, the algorithms are only suited for two-class problems.

Three new methods are proposed here, based on the full expression of the criteria, and on less complex solving process. We will show that these methods have comparable or better results than existing methods, at a generally much lower cost, and can easily be extended to multiclass problems. The next section describes the aforementioned criteria and the proposed feature selection methods.

Notations:

$\mathbf{1}_k, \mathbf{1}$	denotes a $k \times k$ unit matrix filled with 1.
$\boldsymbol{A} \circ \boldsymbol{B}$	entry-wise product of two matrices or vectors.
$\langle \boldsymbol{A}, \boldsymbol{B} \rangle_F$	Frobenius inner product of two matrices.
$\|\|\boldsymbol{A}\|\|_F$	the corresponding norm.
$\Sigma(\boldsymbol{A})$	sum of all entries of matrix ($\Sigma(\boldsymbol{A}) = \sum_{i,j} a_{ij}$).
$\partial_\theta x$	partial derivative of x with regard to θ (i.e. $\frac{\partial x}{\partial \theta}$).
$\boldsymbol{w_h}$	denotes the hyperplane normal vectors.
$\boldsymbol{w}$	denotes the scale factors of a scaled kernel $k_{\boldsymbol{w}}$.

III. Kernel Target Alignment

A. Definition of the Alignment

We consider here a kernel k and a training set $\mathcal{S} = \{(\boldsymbol{x_i}, y_i)\}_{i=1\dots n}$ with $y_i \in \{+1; -1\}$ ($\boldsymbol{y} = [y_1 \dots y_n]^T$), defining a two-class problem. Without loss of generality, the examples are ordered such that the n_1 first belong to $\mathcal{S}_1 = \{(\boldsymbol{x_i}, y_i), y_i = +1\}$ and the n_2 last to $\mathcal{S}_2 = \{(\boldsymbol{x_i}, y_i), y_i = -1\}$, with $n = n_1 + n_2$.

The *Gram matrix* $\boldsymbol{K}$ related to k and $\mathcal{S}$, is defined as $[\boldsymbol{K}]_{ij} = k_{ij} = k(\boldsymbol{x_i}, \boldsymbol{x_j})$. Let $\boldsymbol{K^*} = \boldsymbol{y}\boldsymbol{y^T}$ be the ideal Target matrix. These can be decomposed as class-wise kernel matrices,

$$\boldsymbol{K} = \begin{pmatrix} \boldsymbol{K_{11}} & \boldsymbol{K_{12}} \\ \boldsymbol{K_{21}} & \boldsymbol{K_{22}} \end{pmatrix} \qquad \boldsymbol{K}^* = \begin{pmatrix} \mathbf{1} & \mathbf{-1} \\ \mathbf{-1} & \mathbf{1} \end{pmatrix} \tag{2}$$

In [27], Cristianini et al. introduce a new criterion measuring the similarity between $\boldsymbol{K}$ and $\boldsymbol{K^*}$, called *Kernel Target Alignment* (KTA), based on the Frobenius inner product of two matrices, defined as

$$\langle \boldsymbol{A}, \boldsymbol{B} \rangle_F = \sum_i \sum_j a_{ij} b_{ij} = \Sigma(\boldsymbol{A} \circ \boldsymbol{B}).$$

The Alignment $\mathcal{A}(\boldsymbol{K}, \boldsymbol{K^*})$ is the normalized Frobenius inner product between the Gram matrix and the Target matrix:

$$\mathcal{A}(\boldsymbol{K}, \boldsymbol{K^*}) = \frac{\langle \boldsymbol{K}, \boldsymbol{K^*} \rangle_F}{||\boldsymbol{K^*}||_F ||\boldsymbol{K}||_F} = \frac{\langle \boldsymbol{K}, \boldsymbol{K^*} \rangle_F}{n\ ||\boldsymbol{K}||_F}. \tag{3}$$

The value $1 - \mathcal{A}$ is proved to be an upper bound of the generalization error of the Parzen window estimator [27]. Maximizing the KTA therefore tunes the kernel for the discrimination task described by the training set $\mathcal{S}$. The word *Alignment* will further refer to the Kernel Target Alignment.

In the case of uneven class sets ($n_1 \neq n_2$), the Target matrix can be adapted by compensating the proportions of both classes [31], i.e. $\boldsymbol{\hat{K}^*} = \boldsymbol{\hat{y}}\boldsymbol{\hat{y}^T}$ with $\hat{y}_i \in \{\frac{1}{n_1}; \frac{-1}{n_2}\}$, which can be decomposed in class-homogeneous blocks:

$$\boldsymbol{\hat{K}^*} = \frac{1}{n_1 n_2} \begin{pmatrix} \frac{n_2}{n_1}\mathbf{1} & \mathbf{-1} \\ \mathbf{-1} & \frac{n_1}{n_2}\mathbf{1} \end{pmatrix} \quad \text{with} \quad ||\boldsymbol{\hat{K}^*}||_F = \frac{n}{n_1 n_2} \tag{4}$$

Another way of looking at the Alignment expression emerges from its numerator (the Frobenius inner product) with the inner products expression of the kernels:

$$\left\langle \boldsymbol{K}, \boldsymbol{\hat{K}^*} \right\rangle_F = \left|\left| \frac{1}{n_1} \sum_{x_1 \in \mathcal{S}_1} \Phi(\boldsymbol{x_1}) - \frac{1}{n_2} \sum_{x_2 \in \mathcal{S}_2} \Phi(\boldsymbol{x_2}) \right|\right|^2 \tag{5}$$

Maximizing the Frobenius inner product is thus equivalent to maximizing the inter-cluster distance, or *between-class scatter*, in the feature space, as applied in [32] for kernel hyper-parameters tuning. Note that the Frobenius norm, used in the Alignment denominator, is not geometrically interpretable in the feature space since it involves squared inner products in the feature space ($||\boldsymbol{K}||_F^2 = \sum_{i,j} \langle \Phi(\boldsymbol{x_i}), \Phi(\boldsymbol{x_j}) \rangle^2$). However, as stated in [27], the alignment measure is both proportional to the within-class similarity and inversely proportional to the between-class similarity. Those considerations show the close relationship between the Alignment measure and the classical scatter-based Class Separability measure.

B. Method 1: Scaled Alignment Selection (SAS)

We propose here to perform an iterative maximization of the Alignment $\mathcal{A}$ through a simple gradient ascent on the scaling factors w_i of the *scaled kernel* $k_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}) = k(\boldsymbol{w} \circ \boldsymbol{x}, \boldsymbol{w} \circ \boldsymbol{y})$.

The features are ranked after maximization by descending scale factor order, assuming that the most weighted features contribute the most to the decision function. This approach, based on a KPO search strategy, is named *Scaled Alignment Selection* (SAS), and is summed up in Algorithm 1.

Algorithm 1 Method 1: Scaled Alignment Selection (SAS)

```
$\boldsymbol{\theta}_0 = [1, \ldots, 1]\,, \quad \mathcal{A}_0 = 0, \quad n = 0$
$\boldsymbol{K}^* = \boldsymbol{y}\boldsymbol{y}^T$
**repeat**
   $n \leftarrow n + 1$
   Gram-matrix computation: $[\boldsymbol{K}_{\boldsymbol{\theta}}]_{ij} = k_\theta(\boldsymbol{x}_i, \boldsymbol{x}_j)$
   Parameters update: $\boldsymbol{\theta}_n \leftarrow \boldsymbol{\theta}_{n-1} + \eta\, \partial_{\boldsymbol{\theta}} \mathcal{A}_{n-1}(\boldsymbol{K}_{\boldsymbol{\theta}}, \boldsymbol{K}^*)$
**until** convergence: $|\mathcal{A}_n - \mathcal{A}_{n-1}| < \epsilon$
```

Note that in the case of the RBF Gaussian kernel, σ is implicitly fitted through the scale factors. Indeed, if $\boldsymbol{\theta} = (\sigma, \boldsymbol{w})$, let $\tilde{\sigma}$ be an arbitrary value, and $\tilde{\boldsymbol{\theta}} = \left(\tilde{\sigma}, \frac{\tilde{\sigma}}{\sigma}\boldsymbol{w}\right)$, then

$$k_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\frac{\sum_d \left(\frac{\tilde{\sigma}}{\sigma} w_d\right)^2 (x_d - y_d)^2}{2\tilde{\sigma}^2}\right) = k_{\tilde{\boldsymbol{\theta}}}(\boldsymbol{x}, \boldsymbol{y}). \tag{6}$$

The Alignment derivation is quite straightforward, since it is only additive. It can therefore be used for kernel tuning, based on a gradient ascent [33]. Considering a kernel $k_{\boldsymbol{\theta}}$ characterized by $\boldsymbol{\theta} = (\theta_1 \ldots \theta_P)$, and $\boldsymbol{K}_{\boldsymbol{\theta}}$ its corresponding Gram matrix, one shows that

$$\partial_{\theta_p} \langle \boldsymbol{K}_{\boldsymbol{\theta}}, \boldsymbol{K}^* \rangle_F = \left\langle \partial_{\theta_p} \boldsymbol{K}_{\boldsymbol{\theta}}, \boldsymbol{K}^* \right\rangle_F \tag{7}$$

$$\partial_{\theta_p} ||\boldsymbol{K}_{\boldsymbol{\theta}}||_F = \frac{\left\langle \partial_{\theta_p} \boldsymbol{K}_{\boldsymbol{\theta}}, \boldsymbol{K}_{\boldsymbol{\theta}} \right\rangle_F}{||\boldsymbol{K}_{\boldsymbol{\theta}}||_F}, \tag{8}$$

where we have defined $\partial_{\theta_p} \boldsymbol{K}_{\boldsymbol{\theta}} = [\partial_{\theta_p} k_\theta(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{ij}$. The Alignment can then be derivated with respect to any set of parameters $\boldsymbol{\theta}$. The derivation only involves the additional computation of the matrices $\partial_{\theta_p} \boldsymbol{K}_{\boldsymbol{\theta}}$:

$$\partial_{\theta_p} \mathcal{A}(\boldsymbol{K}_{\boldsymbol{\theta}}, \boldsymbol{K}^*) = \frac{\left\langle \partial_{\theta_p} \boldsymbol{K}_{\boldsymbol{\theta}}, \boldsymbol{K}^* \right\rangle}{||\boldsymbol{K}_{\boldsymbol{\theta}}||||\boldsymbol{K}^*||} - \frac{\langle \boldsymbol{K}_{\boldsymbol{\theta}}, \boldsymbol{K}^* \rangle \left\langle \boldsymbol{K}_{\boldsymbol{\theta}}, \partial_{\theta_p} \boldsymbol{K}_{\boldsymbol{\theta}} \right\rangle}{||\boldsymbol{K}_{\boldsymbol{\theta}}||^3 ||\boldsymbol{K}^*||} \tag{9}$$

C. A note on Sparsity

As stated in the introduction, sparsity is a common goal in feature selection. In the weighted kernel context this means the zeroing of a large proportion of the weights w_i. However, in the presence of redundant features, sparsity doesn't explicitly emerges from the optimization. For this reason, many strategies were proposed in the literature to force the rejection of features. For instance, [6] and [16] both perform iterative exclusion of the least-ranked features in each step of the optimization loop. As a result, as stated in [15], feature selection, when coupled with such strategies, can both be designed for a specified number of selected features S, or as a way to determine an optimal set, under a certain stop condition.

The remainder of this paper, and especially the experiments, will only deal with the first of these two scenarios.

D. Feature-wise derivative matrices

Once the Gram matrix $\boldsymbol{K}$ is computed, the Alignment derivation involved in the SAS method only requires the additional computation of the *feature-wise derivative* matrices $\partial_{w_d} \boldsymbol{K}_{\boldsymbol{w}}$ (denoted $\partial_d \boldsymbol{K}$ here for convenience). We provide here the development for some common kernels:

- **Linear:**
  $k_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{w} \circ \boldsymbol{x}) \cdot (\boldsymbol{w} \circ \boldsymbol{y}) = \sum_d w_d^2 \kappa^d(\boldsymbol{x}, \boldsymbol{y})$
  $\kappa^d(\boldsymbol{x}, \boldsymbol{y}) = x_d \cdot y_d$
  $\partial_d k_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}) = 2\, w_d\, \kappa^d(\boldsymbol{x}, \boldsymbol{y})$
- **Gaussian RBF:**
  $k_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(\frac{-||\boldsymbol{w} \circ (\boldsymbol{x} - \boldsymbol{y})||^2}{2\sigma^2}\right) = \exp\left(-\sum_d w_d^2 \kappa^d(\boldsymbol{x}, \boldsymbol{y})\right)$
  $\kappa^d(\boldsymbol{x}, \boldsymbol{y}) = (2\sigma^2)^{-1}(x_d - y_d)^2$
  $\partial_d k_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}) = -2\, w_d\, \kappa^d(\boldsymbol{x}, \boldsymbol{y})\; k_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y})$
- **Polynomial:**
  $k_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}) = \chi_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y})^\delta$
  $\chi_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}) = 1 + c\,(\boldsymbol{w} \circ \boldsymbol{x}) \cdot (\boldsymbol{w} \circ \boldsymbol{y}) = 1 + c \sum_d w_d^2 \kappa^d(\boldsymbol{x}, \boldsymbol{y})$
  $\kappa^d(\boldsymbol{x}, \boldsymbol{y}) = x_d \cdot y_d$
  $\partial_d k_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}) = 2\, \delta\, c\, w_d\, \kappa^d(\boldsymbol{x}, \boldsymbol{y})\, \chi_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y})^{\delta - 1}$

These relations show that the $\partial_d \boldsymbol{K}$ matrices can be easily evaluated from the entry-wise product of the Gram matrix and the *feature-wise* matrices $[\boldsymbol{\kappa}^d]_{ij} = \kappa^d(\boldsymbol{x}_i, \boldsymbol{x}_j)$, especially in the case of the linear and RBF kernels. The calculation of the feature-wise matrices at the start of the algorithm thus provides a great increase in computational cost, if enough memory is available. Otherwise, the feature-wise matrices must be computed at each iteration.

The decomposition of the Gram matrix into feature-wise matrices highlights the strong link of the SAS method with the GMKL [23] and SimpleMKL [20] algorithms. Indeed, the latter consist in optimizing a product or a sum of feature-wise kernels through an optimization loop involving scale factors. The main difference lies in the choice of the optimization criterion: SAS is based on the Alignment, while the two others lies on SVM trainings, i.e. margin maximization.

E. Extension to multiclass problems

The Alignment measure can also be extended to multiclass problems[1], involving a number of C classes. As proposed by Vert [34], this is done by simply defining the following Target kernel:

$$k_C^*(\boldsymbol{x}_i, \boldsymbol{x}_j) = \begin{cases} 1 & \text{if} \quad y_i = y_j \\ -1/(C-1) & \text{if} \quad y_i \neq y_j \end{cases}. \tag{10}$$

This defines a valid kernel, which is not the case with the naive choice of $k_C^*(\boldsymbol{x}_i, \boldsymbol{x}_j) = -1$ if $y_i \neq y_j$. The corresponding Gram matrix is substituted to the Target matrix defined in equation 2. For instance, for $C = 3$ classes:

$$\boldsymbol{K}_3^* = \begin{pmatrix} \mathbf{1} & -\frac{1}{2}\mathbf{1} & -\frac{1}{2}\mathbf{1} \\ -\frac{1}{2}\mathbf{1} & \mathbf{1} & -\frac{1}{2}\mathbf{1} \\ -\frac{1}{2}\mathbf{1} & -\frac{1}{2}\mathbf{1} & \mathbf{1} \end{pmatrix}. \tag{11}$$

The rest of the theory, e.g. equations 3, 7 and 9, holds true with this new Target matrix, the SAS algorithm is thus basically the same on a multiclass problem. The multiclass notation will hold in the following section, since the next measure is directly defined on that case.

[1]By multiclass we denote problems that involve more than two classes.

IV. Kernel Class Separability

A. Introduction of the Kernel trick

Several measures exist in the literature to evaluate class separability [35], among which the scatter-matrix-based is the most common. We have pointed in section III-A the close relationship between the KTA and the between-class scatter in the feature space. The expression of the full scatter-matrix measure in the feature space has been explored [36] [37] [8], and is usually compared with the KTA. The criterion generally considered involves the *between-class* and *within-class scatter matrices* $\boldsymbol{S}_b$ and $\boldsymbol{S}_w$:

$$\boldsymbol{S}_b = \frac{1}{n} \sum_{c=1\ldots C} n_i (\boldsymbol{m_c} - \boldsymbol{m})(\boldsymbol{m_c} - \boldsymbol{m})^T \quad (12)$$

$$\boldsymbol{S}_w = \sum_{c=1\ldots C} \sum_{\boldsymbol{x}\in\mathcal{S}_c} (\boldsymbol{x} - \boldsymbol{m_c})(\boldsymbol{x} - \boldsymbol{m_c})^T, \quad (13)$$

where $\boldsymbol{m_c}$ denotes the center of class c ($\boldsymbol{m_c} = \frac{1}{n_c}\sum_{\boldsymbol{x}\in\mathcal{S}_c} \boldsymbol{x}$), and $\boldsymbol{m}$ the center of all examples ($\boldsymbol{m} = \frac{1}{n}\sum_{\boldsymbol{x}\in\mathcal{S}} \boldsymbol{x} = \frac{1}{n}\sum_c n_c \boldsymbol{m_c}$). The scatter-based class separability measure takes the following expression:

$$\mathcal{C} = \frac{\operatorname{tr} \boldsymbol{S}_b}{\operatorname{tr} \boldsymbol{S}_w}. \quad (14)$$

Note that the determinant can be used instead of the trace but the latter simplifies computation, and remains more stable in presence of nearly singular matrices. This measure will also been found in Fisher Discriminant Analysis (Section IV-D).

The Kernel trick can be introduced [37] in order to estimate the class separability measure in the feature space. Let the operator Σ be the sum of a matrix entries ($\Sigma \boldsymbol{M} = \sum_{i,j} m_{ij}$), the terms in the feature space can be expressed as follows,

$$\operatorname{tr} \boldsymbol{S}_b = \Sigma \boldsymbol{W} - \frac{1}{n}\Sigma \boldsymbol{K} \quad (15)$$

$$\operatorname{tr} \boldsymbol{S}_w = \operatorname{tr} \boldsymbol{K} - \Sigma \boldsymbol{W}, \quad (16)$$

with

$$\boldsymbol{W} = \frac{1}{n} \begin{pmatrix} \frac{1}{n_1}\boldsymbol{K_{11}} & & \boldsymbol{0} \\ & \ddots & \\ \boldsymbol{0} & & \frac{1}{n_C}\boldsymbol{K_{CC}} \end{pmatrix}, \quad (17)$$

where the $\boldsymbol{K_{ij}}$ blocks are the class-wise submatrices introduced in section III-A, equation 2. The Kernelized Class Separability (KCS) measure $\mathcal{C}_K$ then equals:

$$\mathcal{C}_K = \frac{\Sigma \boldsymbol{W} - \frac{1}{n}\Sigma \boldsymbol{K}}{\operatorname{tr} \boldsymbol{K} - \Sigma \boldsymbol{W}} \quad (18)$$

B. Relation with the Alignment criterion

The *Kernel Class Separability* (KCS) expression is somehow close to that of the KTA. Indeed, in the two-class case, with the matrix $\hat{\boldsymbol{K}}^*$ defined for uneven class sets (eq. 4),

$$||\hat{\boldsymbol{K}}^*||_F = \frac{n}{n_1 n_2} \quad (19)$$

$$\left\langle \boldsymbol{K}, \hat{\boldsymbol{K}}^* \right\rangle_F = \frac{n}{n_1 n_2} \operatorname{tr} \boldsymbol{S}_b. \quad (20)$$

The Alignment then equals

$$\mathcal{A}(\boldsymbol{K}, \hat{\boldsymbol{K}}^*) = \frac{<\boldsymbol{K}, \hat{\boldsymbol{K}}^*>_F}{||\hat{\boldsymbol{K}}^*||_F ||\boldsymbol{K}||_F} = \frac{\operatorname{tr} \boldsymbol{S}_b}{||\boldsymbol{K}||_F} \quad (21)$$

The numerator is thus common in both criteria. Nevertheless, the denominators differ between the two. In the KCS, only the same-class inner products are considered, weighted by their class representation in the training set, whereas in the KTA, all the squared products of the examples are equally summed. Hence, while the KCS normalization aims at minimizing the intra-class scatter, the KTA normalization results in minimizing the global scatter, regardless of the examples' classes.

C. Method 2: Scaled Class Separability Selection (SCSS)

Similarly to the Alignment-based approach, the KCS criterion can be derivated with respect to the kernel scale factors. The derivation of $\operatorname{tr} S_b$ and $\operatorname{tr} S_w$ is straightforward from equations 15 and 16:

$$\partial_\theta \operatorname{tr} S_b = \Sigma(\partial_\theta \boldsymbol{W}) - \frac{1}{n}\Sigma(\partial_\theta \boldsymbol{K}), \quad (22)$$

$$\partial_\theta \operatorname{tr} S_w = \operatorname{tr} \partial_\theta \boldsymbol{K} - \Sigma(\partial_\theta \boldsymbol{W}). \quad (23)$$

However, while being more reliable, because of its interpretation in the feature space, the expression of the KCS induces numerical instability in the maximization of $J(\boldsymbol{w}_h)$. Indeed, when coupled with a RBF Gaussian kernel, both scatter measures converge to zero ($\operatorname{tr} \boldsymbol{S}_b \to 0$ and $\operatorname{tr} \boldsymbol{S}_w \to 0$ when $w_d \to 0$), leading the KCS measure to converge to its upper bound ($\mathcal{C}_K \to 1$). This is prevented here by adding a regularization term to the denominator:

$$\tilde{\mathcal{C}}_K = \frac{\operatorname{tr} \boldsymbol{S}_b}{\operatorname{tr} \boldsymbol{S}_w + \epsilon}$$

The feature selection algorithm proposed here, based on the KCS criterion with a KPO search strategy on the scale factors is called *Scaled Class Separability Selection* (SCSS).

Wang also proposed [8] a feature selection scheme based on the KCS. However, in order to bypass the regularization issue, the author states that the KCS criterion is lower-bounded by $\operatorname{tr} \boldsymbol{S_b}$ and thus bases all his experiments on this latter criterion, which is in fact the sole Frobenius criterion (equations 15 and 20) in which the class variances are not expressed. Our experiments will show that the full KCS criterion, though not as efficient as the Alignment in practice, is more reliable than the Frobenius criterion.

D. Method 3: Kernel Fisher Discriminant Selection (KFDS)

The scatter-matrix class separation is also related to the Kernel Fisher Discriminant Analysis (KFDA). FDA consists in the evaluation of a vector $\boldsymbol{w}_h$ that determines the optimal discriminative hyperplane between two classes. The kernelized problem is expressed in [36] as the maximization of the value $\mathcal{J}(\boldsymbol{w}_h)$ in the feature space,

$$\mathcal{J}(\boldsymbol{w}_h) = \frac{\boldsymbol{w}_h^T S_b \boldsymbol{w}_h}{\boldsymbol{w}_h^T S_w \boldsymbol{w}_h}.$$

The Reproducing Kernels theory states that the vector $\boldsymbol{w}_h$ necessary lies in the span of all training examples, i.e. $\boldsymbol{w}_h = \sum_i \alpha_i \Phi(\boldsymbol{x}_i)$. This allows the expression of $\mathcal{J}(\boldsymbol{w}_h)$ in terms of inner products and thus its *kernelization*, based on the Gram matrix $\boldsymbol{K}$. The problem reformulated with the α_i becomes

$$\mathcal{J}(\boldsymbol{\alpha}) = \frac{\boldsymbol{\alpha}^T \boldsymbol{M} \boldsymbol{\alpha}}{\boldsymbol{\alpha}^T \boldsymbol{N} \boldsymbol{\alpha}}$$

where $\boldsymbol{M}$ and $\boldsymbol{N}$ are both expressed from $\boldsymbol{K}$, and $\boldsymbol{\alpha} = [\alpha_1 \dots \alpha_n]^T$. The problem is solved analogously to the linear algorithm (i.e. without kernel products), by finding the leading eigenvector of $\boldsymbol{N}^{-1}\boldsymbol{M}$. Please consult [36] for more details on the Kernel Fisher Discriminant Analysis.

The *Kernel Fisher Discriminant Selection* (KFDS) approach we propose here, consists in using the hyperplane normal vector $\boldsymbol{w}_h$, determined with KFDA, to iteratively update the scale factors of the kernel. It is similar to AROM, and implies the same kernel restriction (explicit Φ transform to evaluate $\boldsymbol{w}_h$), but implies a matrix inversion instead of the SVM optimization loop to evaluate the vector $\boldsymbol{w}_h$. KFDA is detailed in the Alg. 2 below.

Algorithm 2 Method 3: Kernel Fisher Discriminant Selection

```
Scale factors: $\boldsymbol{w}^1 = [1, \dots, 1]$
$n \leftarrow 0$
repeat
    $n \leftarrow n + 1$
    Compute scaled Gram-matrix: $[\boldsymbol{K}_{\boldsymbol{w}^n}]_{ij} = k_{\boldsymbol{w}^n}(\boldsymbol{x}_i, \boldsymbol{x}_j)$.
    Compute $\boldsymbol{M}$ and $\boldsymbol{N}$ from $\boldsymbol{K}_{\boldsymbol{w}^n}$, and $\boldsymbol{\alpha}^n = \boldsymbol{M}/\boldsymbol{N}$.
    New normal vector: $\boldsymbol{w}_h^n = \sum_i \alpha_i^n \boldsymbol{x}_i$.
    Scale factors update: $\boldsymbol{w}^{n+1} = \boldsymbol{w}^n \circ \boldsymbol{w}_h^n$.
until convergence on scale factors $\boldsymbol{w}^n$
```

V. Numerical experiments

We compare here the proposed methods to existing kernel-based feature selection methods (the ones available in the Spider machine learning toolbox), both on synthetic and real world data (respectively Sections V-A and V-B). We give here a brief review of the methods.

Proposed methods:

- 1) *Scaled Alignment Selection (SAS)*: see section III-B.
- 2) *Scaled Class Separability Selection (SCSS)*: see IV-C.
- 3) *Kernel Fisher Discriminant Selection (KFDS)*: see section IV-D. Only used with a linear kernel.

Reference methods:

- *Fisher*: Fisher criterion.
- *AROM*: approximation of the zero-norm minimization with a L2 norm [15]. Only used with a linear kernel.
- *R2W2*: gradient descent based on the radius-margin criterion to estimate the scale factors [6].
- *RFE*: backward sequential selection based on the ranking of the hyperplane normal vector components [16].

We will comment on graphic curves for their readability. On all figures, the proposed methods are indicated with dotted lines, while solid lines are used for existing methods.

The SFS test-method will be commented in Section V-C, which provides other experiments that intend to compare with the published results of similar methods. Because we could not implement these methods, we only reproduced the protocol of various experiments involving the latter. As explained in Section V-C, SFS cannot be used with a linear kernel since the weight factors diverge to infinity. Therefore, results are only shown with non-linear kernels.

Student tests with $\alpha = 5\%$ significance level were also evaluated in each experiment between all pairs of methods, to assess the significance of performance gaps. They cannot all be shown here but will be mentioned in a few experiments.

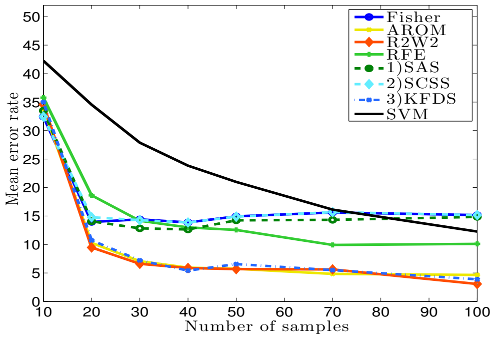
In order to show reproducible research, we publicly provide the source code of the methods proposed here, along with further details on the features of the speech/music dataset introduced later on, at http://www.mathieuramona.com/wp/data/fsgram.

A. Toy Experiments

The synthetic data, as well as the experimental protocol, are drawn from the experiments detailed in [6] and [17]: those compare the performance on linearly separable and non-separable problems (respectively trained with a linear and a Gaussian RBF kernel) through the evaluation on two features selected among a large set of irrelevant or redundant features. The linear problem gathers 202 features of which 6 are relevant but redundant, based on Gaussian distributions, and the rest are noise. The non-linear problem gathers 52 features of which only 2 are relevant, but draw a linearly inseparable distribution.

n training examples are randomly drawn (n ranging from 10 to 100). The best two features are selected and an SVM is trained with those two on the same training set. The average test error is computed on a test set of 500 samples drawn from the same distribution at each iteration, over 40 iterations of training and testing. The results are shown in figure 1. The results with an SVM training on the whole set of features are also shown (on solid black lines), to evaluate the gain brought by the feature selection step.

The linear problem, shown on the upper Fig. 1(a), illustrates the main drawback of the proposed scaled methods in failing to find the best solution within strongly redundant features. The scores converge around a 15% error rate while n increases, along with the Fisher criterion, because two relevant but redundant features are selected, instead of the two complementary features. R2W2, AROM and the proposed KFDS succeed on this problem, with very close results. The second problem (lower Fig. 1(b)) focuses on the inter-relevance of the two non-noisy features. With no surprise, the Fisher criterion fails here, along with the linear-kernel methods (RFE, AROM and KFDS). R2W2 approach here shows the best behavior. The KCS approach (SCSS) here shows more efficient than KTA-based (SAS), but this is no longer observed on real world data. R2W2, SCSS, SAS and KFDS are all significantly different, according to the 5%-level Student test.



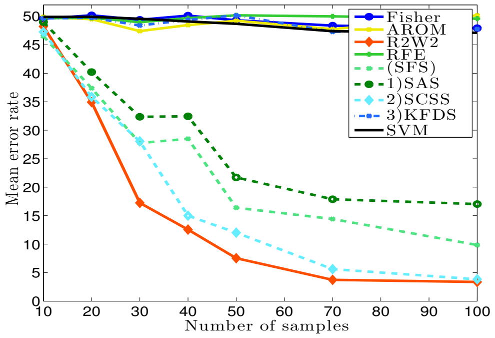


Fig. 1. Comparison of performances on a linearly separable problem (a, upper) involving redundant features and a non-linearly separable problem (b, lower). Both contain many irrelevant noisy features.

B. Real World Data

The proposed methods were tested on various real world datasets, representing different sorts of configurations. *Ionosphere*, *Spambase*, *Parkinsons*, *Sonar*, *Liver*, *BCW*, *Cleveland*, *Pima* are all from the UCI public repository [38]. Two gene expression microarray datasets (*Lymphoma* and *Yeast*) are retrieved from Weston [15] [16], characterized by very few training examples. *Lymphoma* has a very large feature collection (several thousands). A dataset from our research in audio indexing (speech/music discrimination) is also provided and freely available[2]. The multiclass digit recognition dataset USPS is retrieved from LibSVM[3].

This section only implies reference methods that could be reproduced, and compares with our methods. Further experiments, Section V-C, will compare our results with the results of various publications.

The feature selection is operated on a training subset of size n_{train} from the dataset (which contains n_1 and n_2 from each class, respectively). An SVM is then trained on the same subset over a varying range of the S best ranked among the D features. The penalty factor C is always set to the estimated optimal value proposed by Joachims[4] in the implementation of *SVMlight* [39], including inside the R2W2, RFE and AROM loops (several experiments conducted internally confirm that this $\hat{C}$ value induces the minimization of the Leave-One-Out error). The error rate is computed by applying the trained SVM on a test subset of size n_{test}, also extracted from the dataset, but distinct from the training set.

[2]http://www.mathieuramona.com/wp/data/fsgram

[3]http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

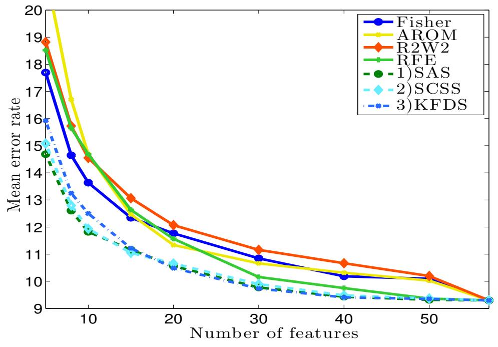
[4]$\hat{C} = 1/\bar{R}^2$, where $\bar{R} = \sum_{i=1}^{n} ||\phi(\boldsymbol{x}_i) - \phi(\mathbf{0})||$ is an estimation of the radius of the smallest sphere containing all examples in the feature space.

Data set	C	D	Size n_1, n_2	n_{train}	n_{test}
Spambase	2	57	2788 / 1813	500	1000
Ionosphere	2	34	225 / 126	250	101
Lymphoma	2	4026	34 / 62	60	36
Speechmusic	2	321	2× 10000	500	500
USPS	10	256	7291	10×100	2007
Yeast	5	79	208	182	26
Parkinsons	2	22	48 / 147	136	59
Used only in Section V-C:					
BCW	2	9	444 / 239	340	343
Cleveland	2	13	160 / 137	150	147
Liver	2	6	145 / 200	240	105
Musk	2	166	207 / 269	300	176
Pima	2	8	500 / 268	500	268
Sonar	2	60	111 / 97	145	63

TABLE I
COMPARED CHARACTERISTICS OF THE DATA SETS.

The mean error rate is then evaluated over 30 iterations. We provide results for both linear and non-linear (RBF) kernels in most cases. Table I sums up the dataset characteristics.

1) Spambase & Ionosphere: Both sets are from the UCI repository [38] and are tested with linear and RBF kernels.



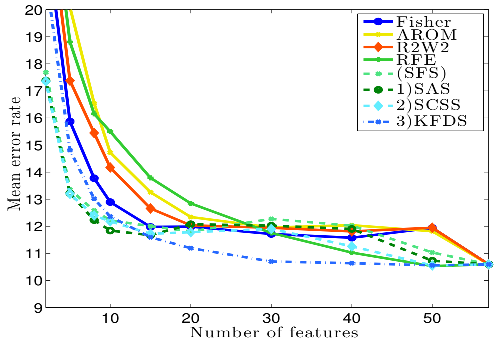


Fig. 2. Compared performances on the *Spambase* dataset, with a linear (upper figure) and a Gaussian RBF (lower figure) kernel.

The results on the *Spambase* set, Figure 2, assess the linear separability of the problem since RBF kernel brings no improvement. Irrelevant features do not seem to penalize classification, since the error rate increases with the selection. On both kernels, the proposed algorithms (in dotted lines) clearly prove more efficient than existing ones, providing up to almost a 3% error decrease with 10 features on a linear kernel, compared to the 15% error rate with R2W2 (i.e. a relative error reduction of 20%). The KFDS approach, though based on a linear kernel, shows surprisingly good performance when followed by a SVM with RBF kernel. The RBF kernel experiment also confirms the relevance of keeping the Alignment normalization with the SAS approach, when compared to the Scaled Frobenius (SFS).

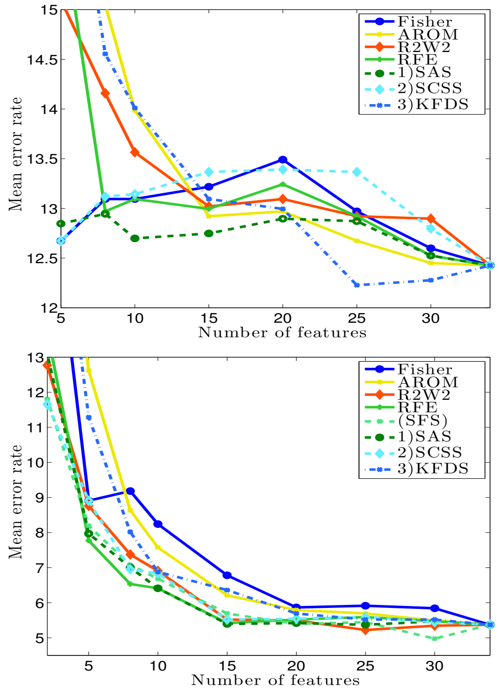


Fig. 3. Compared performances on the *Ionosphere* dataset, with a linear (upper figure) and a Gaussian RBF (lower figure) kernel.

On the opposite, *Ionosphere* (Figure 3) is clearly linearly non-separable. The error rate remains above 12.5% with a linear kernel while the RBF kernel reduces error to nearly 5% (please note the difference of ordinates scale between sub-figures). The poor performance of the linear-based approaches (Fisher, KFDS and AROM) with a RBF Gaussian kernel, confirms this observation. We thus focus on the RBF kernel case (lower figure). The results confirm the observations on *Spambase*: a slight decrease of performance with SFS, when compared to SAS ; and no gain over Scaled Alignment (SAS) when using the scaled Class Separability (SCSS). Nevertheless, both methods provide comparable performance with R2W2. The 5% level Student test shows no significant difference between the two.

2) Lymphoma microarray: DNA microarray data analysis generally involves very small data sets (built from human cases) with a large number of gene-based features. The Lymphoma problem reproduces the experiment described in [15] (originally proposed for the AROM method evaluation). This example tests the reliability on very large collections of features, possibly highly redundant. RFE is not tested here because of its very high complexity, quadratic with the number of features. Following the original protocol, this problem is tested with a linear kernel. Results are shown in Figure 4.

The error decrease with R2W2, KFDS and AROM attests the presence of many irrelevant features. The selection strongly improves the performance for some methods, with very few selected features. Indeed, the information brought

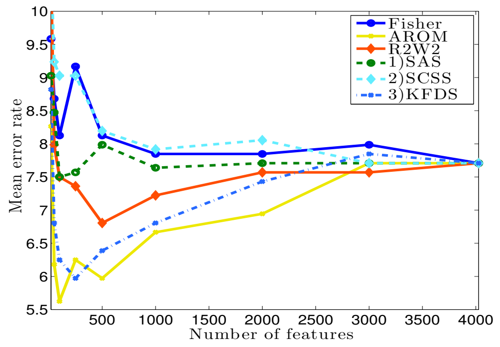


Fig. 4. Performance on the *Lymphoma* dataset, with a linear kernel.

by the useful features is easily diluted by the admixture of non-relevant information from the other features. Moreover, the bad performance of the Fisher criterion is an evidence of the strong interdependence of the features. The experiment shows the limit of the scaled kernel methods proposed here (SAS and SCSS) in dealing with very large feature collections. However, the KFDS method, if not as efficient as AROM, succeeds very well in increasing the performance (about 1.5% of error decrease), and outperforms R2W2. Student test shows that all methods are significantly different, with a 5% significance level.

3) Audio indexing: Speech/Music: The last dataset is built from our works on audio indexing and describes a speech/music discrimination problem on broadcast news extracts. Its 321 features describe different acoustic properties (temporal, spectral, cepstral and perceptual). Each class contains 10000 samples. Please consult our previous work [40] for more details, or http://www.mathieuramona.com/wp/data/fsgram where the dataset is freely available.

The results with a linear kernel, on the upper figure 5, do not indicate the effect of noisy features at high dimensions, but the low slope above 100 features, reflects the strong redundancy between them. Redundant features can be interpreted as a unique overscaled feature, that, amplified by the RBF kernel exponentiation, penalizes the classifier performance. This is clearly visible on the lower figure 5, where most methods show a neat error rate decrease. These strong redundancies explain the weak performance of the Fisher criterion with linear kernel. While the R2W2 method shows the best performance, the proposed Scaled Alignment (SAS) provide comparable results. The Student test proves that the difference is not significant between the two (with 5% significance level), but shows that the difference with the other methods is significant.

4) Multiclass selection: To demonstrate the effectiveness of SAS and SCSS on multiclass problems, we have adapted the previous protocol to two multiclass datasets.

The first one if the widely used US Postal Services digit recognition which contains 10 classes. Over 30 iterations, a subset of 100 samples per class is used for selection and SVM training, and the rest for testing. A pairwise scheme is used for classification (45 SVMs trained), combined with the

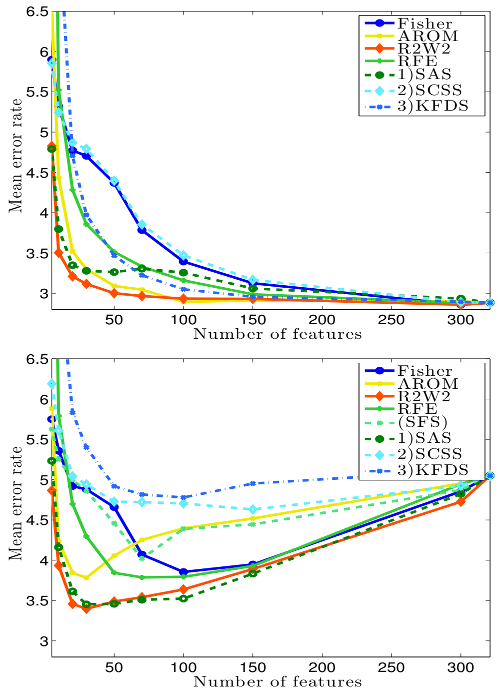


Fig. 5. Compared performances on the audio indexing *Speech/Music* dataset, with a linear (upper figure) and a Gaussian RBF (lower figure) kernel.

Pairwise Coupling algorithm [41], to determine the maximum likelihood class on the posterior probabilities. Results show (Table II), that the two algorithms perform better than Fisher and FSPP methods, although AROM gets the best results.

	Fisher	FSPP	AROM	1)SAS	2)SCSS
200	6.7±1.0	6.6±0.8	6.2±0.8	6.9±1.0	6.6±1.0
150	6.4±1.0	6.9±0.7	6.0±0.8	6.6±1.1	6.7±1.0
100	6.3±1.0	7.1±0.7	5.8±0.8	6.2±1.0	6.6±1.0
50	6.7±1.0	8.0±0.7	5.6±0.8	6.2±1.0	6.5±1.1
30	7.5±1.2	9.1±0.9	5.5±0.8	6.5±1.0	6.9±1.0
10	10.1±1.6	12.4±1.3	6.3±0.7	8.3±1.3	8.9±1.3
5	12.2±1.5	13.8±1.1	8.5±0.7	10.6±1.4	10.8±1.4

TABLE II
MEAN ERROR RATE OVER 30 ITERATIONS ON THE 10-CLASS DATASET USPS WITH GAUSSIAN RBF KERNEL.

The second dataset is the Brown Yeast Microarray described in [15], and used for the evaluation of the AROM method. We have reproduced here the original protocol [15], using an 8-fold cross validation. Results are compared with FSPP and the 1-vs-All multiclass variant of AROM. Table III shows that SAS and SCSS both outperform the reference methods.

	FSPP	AROM	1)SAS	2)SCSS
79	5.6±3.0	5.6±3.0	5.6±3.0	5.6±3.0
40	8.3±3.7	4.6±2.7	3.6±2.5	3.8±2.3
30	8.4±3.6	4.3±2.7	3.3±2.4	3.6±2.2
20	8.7±3.5	3.8±2.8	3.2±2.2	3.0±2.0
10	10.8±4.2	3.8±2.7	3.6±2.3	2.4±2.2
5	13.1±4.0	3.4±2.3	3.8±2.3	2.9±1.9

TABLE III
MEAN ERROR RATE OVER 30 ITERATIONS ON THE 5-CLASS DATASET YEAST WITH GAUSSIAN RBF KERNEL.

5) Polynomial kernel on Parkinsons: The last experiment of this section shows an example of use of the proposed methods with a polynomial kernel (of order 3), instead of the RBF Gaussian, applied on the *Parkinsons* UCI dataset. Results, shows in Figure 6, demonstrate that the SAS and SCSS algorithms remain applicable to any type of kernel, and remain efficient when compared to existing methods. SAS especially outperforms all other methods of this experiment for $S > 6$ (the difference is significant under a 5% Student test).

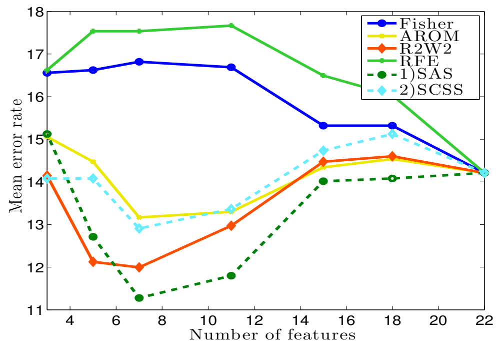


Fig. 6. Compared performances on the *Parkinsons* dataset, using a polynomial kernel (of order $d = 3$).

C. *Comparison with existing Kernel-based methods*

Neumann KTA on UCI datasets [7]

As stated earlier, Neumann et al. also proposed a method involving the Alignment criterion, based on the joint minimization of the Alignment and the zero-norm of the hyperplane vector $\boldsymbol{w_h}$. The objective function is decomposed as a difference of two convex functions, and then minimized with a specific algorithm (DCA, *Difference of Convex functions minimization Algorithm* [42]). The minimization is done through a double-nested loop where each inner step implies the computation of the Alignment, whereas SAS only relies on a simple gradient ascent loop. Moreover the denominator of the Alignment is discarded to get the convex functions decomposition. The criterion really used in practice is therefore the *Frobenius* criterion, defined as follows:

$$\mathcal{F}(\boldsymbol{K}, \boldsymbol{K}^*) = \langle \boldsymbol{K}, \boldsymbol{K}^* \rangle_F . \tag{24}$$

Previous experiments show the results when substituting the Frobenius criterion to the full Alignment, through the test-method SFS (*Scaled Frobenius Selection*). Figures 2(b), 3(b), and 5 show that the full Alignment method SAS is more efficient than the Frobenius-based SFS. Moreover, the absence of normalization has another drawback when using the linear kernel, because the Frobenius criterion diverges to infinity when the scales factors w_d increase arbitrarily. The maximization cannot be done. For this reason, the Alignment is not used with the linear kernel in the experiments of [7].

To further assess this discussion, the latter experiment [7] (pp. 142–145) is reproduced on the same UCI datasets than used by the authors. It is conducted both on linear and RBF Gaussian kernels (we reproduce respectively the ℓ_2-ℓ_1-SVM and KTA-based results). The number of selected features is automatically fixed by the algorithms ; we thus show the results for the number of features indicated by the authors.

Set	KTA [7]		1)SAS	2)SCSS	3)KFDS
	dim	err			
	Linear kernel				
Liver	6.0	35.1±1.0	34.0 ± 2.0	34.0 ± 2.0	34.0 ± 2.0
Cleve	9.9	16.5 ± 0.5	17.2±2.4	17.1±2.5	24.1±2.9
Ionos	25	13.4±0.3	12.9±2.4	13.4±2.6	12.2 ± 2.3
Pima	6.6	25.1±0.2	24.2 ± 2.0	24.2 ± 1.9	24.4±1.8
BCW	8.7	3.2 ± 0.0	3.5±0.9	3.5±0.9	3.5±0.9
Sonar	50	22.6±0.1	23.3±1.8	23.1±1.6	21.9 ± 1.9
Musk	125	18.3 ± 0.3	20.7±2.5	20.0±2.4	21.4±2.6
	Gaussian RBF kernel				
Liver	2.5	35.4±1.5	36.2±1.0	34.9 ± 2.6	40.4±2.3
Cleve	3.2	23.6±0.3	22.1 ± 3.0	23.0±2.8	27.8±3.1
Ionos	6.6	7.7±0.3	7.0 ± 2.2	7.0 ± 2.4	8.0±2.8
Pima	1.4	27.0±0.2	27.0±2.6	26.1 ± 1.2	32.7±3.2
BCW	2.8	4.2 ± 0.0	4.5±1.1	4.3±1.0	6.1±1.3
Sonar	9.6	27.4±0.6	24.9±3.1	22.2 ± 3.3	31.6±2.9
Musk	41	15.5±0.2	13.8 ± 3.0	14.4±3.3	15.9±2.7

TABLE IV
COMPARED ERROR RATES WITH NEUMANN ET AL.

The results (Table IV) show that the proposed methods have comparable or better results in most cases. The RBF kernel is particularly well handled by SAS and SCSS, when compared to the Neumann KTA method. KFDS only relies on a linear kernel, which explains the mitigate results with RBF kernel (used in the SVM training and classification to evaluate the results). However, the latter shows the best performance on 3 datasets with linear kernel. The proposed methods thus offer a good alternative, for a reduced computational cost.

Wang KCS on the binarized USPS dataset [8]

We reproduce here the experiment developed by Wang [8] (p. 1544, Sec. 5.2) to assess the reliability of his Kernel Class Separability (KCS). As mentioned before, the criterion used is in practice also the Frobenius product of equation 24. Wang uses the USPS digit recognition dataset, and converts it into a two-class problem between classes 0…4 and 5…9. He simulates a small sample by using a subset of $m = 7$ samples of each class, used for the feature selection. The SVM is then trained with 1,000 samples, and the result is evaluated over 7 iterations on the 2007 samples of the USPS predefined test dataset.

Results (in error rates), shown in table V, demonstrate that SAS and SCSS perform better on the whole scale of S values than the KCS algorithm. The R2W2 results, as provided by Wang, are also outperformed.

S	R2W2	KCS	1)SAS	2)SCSS	3)KFDS
50	10.0	11.5	10.0±2.5	14.7±4.0	12.7±3.5
40	10.0	12.5	11.4±2.2	16.3±3.9	13.3±3.3
20	15.5	17.0	14.3±1.1	21.4±4.7	17.0±2.9
10	25.0	23.0	20.7±2.5	24.4±4.1	24.5±2.1
5	34.0	31.5	26.6±2.9	27.5±3.3	29.5±2.8
3	39.5	36.5	30.4±2.4	29.0±2.6	33.3±3.6
1	41.0	41.5	37.6±4.8	36.9±5.1	40.2±3.4

TABLE V
COMPARED ERROR RATES WITH WANG [8] ON BINARIZED USPS DATASET.

GMKL on UCI datasets [23]

To conclude this section, we also reproduce the experiment described by Varma and Babu, on various UCI datasets, to assess the efficiency of their GMKL method (presented earlier). We also show the results of the BAHSIC method [26], indicated by the authors. For each set, 70% of the samples are used for selection and training, the other 30% for testing. The kernel is RBF Gaussian. The accuracy is originally indicated for several fixed number of selected features, that are reproduced here. The results are shown in table VI.

D_S	BAHSIC	GMKL	1)SAS	2)SCSS	3)KFDS
Ionosphere					
5	87.1±3.1	90.9±1.9	92.0 ± 2.3	91.1±2.8	88.7±3.1
10	90.2±3.5	93.7 ± 2.1	93.6±2.1	93.2±2.1	93.1±2.8
15	92.6±2.0	94.1±2.1	94.6 ± 2.0	94.5±2.0	93.6±2.2
Parkinsons					
3	85.2±3.8	86.3±4.1	88.6 ± 4.1	84.9±3.8	85.1±7.7
7	88.5±3.6	92.6 ± 2.9	88.7±3.7	84.2±4.5	85.6±6.0
Sonar					
5	61.1±6.2	74.4 ± 5.1	74.1±4.4	71.6±3.8	65.3±7.7
10	73.1±5.9	80.2 ± 4.9	75.1±3.1	77.8±3.3	68.4±8.9
15	74.7±4.5	80.7 ± 5.5	77.4±4.5	79.5±3.8	73.9±7.9

TABLE VI
COMPARED ACCURACIES WITH GMKL [23] AND BAHSIC [26].

Here, GMKL mostly outperforms the proposed methods, except on the Ionosphere dataset. This case is shown because, as explained earlier, the feature-wise decomposition of the Gram matrix highlights a close connection between our methods and MKL theory. However, GMKL relies on SVM solvers and a more robust objective function. The drawback is its high complexity, when compared to our methods. The following Section VI will show that the R2W2 method, very close to GMKL, is indeed more costly than SAS and SCSS.

D. Ranking of proposed methods

Table VII sums up the percentage among the results where each proposed method is ranked first, second or third. The upper part of the Table concerns the results of Section V-B, and clearly shows that SAS is comparable in ranks with R2W2, even though is reaches a little less often the first rank.

The lower part concerns the results of Section V-C. It shows that SAS is first rank in more than 1 over 3 cases. SCSS is already well ranked, being ranked second at least in more than 60% cases. Finally, KFDS is less efficient than the two previous methods.

Method	1st	2nd	rest
Section V-B Real World Data			
1)SAS	23.8%	23.8%	52.4%
2)SCSS	6.0%	16.7%	77.4%
3)KFDS	14.3%	11.7%	74.0%
R2W2	27.4%	15.5%	57.1%
Section V-C Comparison with existing			
1)SAS	37.9%	34.5%	27.6%
2)SCSS	17.2%	44.8%	37.9%
3)KFDS	6.9%	0.0%	93.1%

TABLE VII
PERCENTAGE OF 1ST AND 2ND RANKS OF PROPOSED METHOD.

VI. COMPUTATIONAL ISSUES

A. Iterative computation

One of the main advantages of the SAS and SCSS methods is their scalability in terms of memory usage. From equations 3, 9, 15, 16, 22 and 23, it can be noticed that only sums involving $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $\partial k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $k(\boldsymbol{x}_i, \boldsymbol{x}_j)^2$ or $\partial k(\boldsymbol{x}_i, \boldsymbol{x}_j)^2$ are needed while computing KTA or KCS. The Gram matrices needn't be expressed all at once, since only the Frobenius sums of their terms are of interest. Each term can be computed iteratively, squared for the $||\boldsymbol{K}||$ norm, and immediately discarded after summing. Moreover, the terms of the Target matrices $\boldsymbol{K}^*$, when multiplied with the terms of $\boldsymbol{K}$, are equivalent to identity or sign changes, and therefore imply no computation (apart from sign changes).And since the Gram matrices are symmetric, only half the non-diagonal terms need be computed and can be doubled in the sums. This results in a half decrease of complexity.

The memory usage is thus almost arbitrarily low, since only a single kernel product need be kept in memory at the same time. The dimension (i.e. the number of feature) is of no importance here since the kernel products are only scalars, independent from the examples' vectors components (we consider that the memory volume of the dataset is negligible, since it is only linear with respect to n, while Gram matrices are quadratic). This is not the case for example for method R2W2, which implies the full expression of the Gram matrix to evaluate the radius R (through quadratic programming).

B. Complexity

The computation of the Gram matrix terms is quadratic with respect to the number of examples N and theoretically sublinear with respect to dimension D (because only a part of the kernel products computation involves all the components). Since each feature implies a partial derivative matrix, besides the main Gram matrix, the complexity for SAS and SCSS is about $O(IDN^2)$, where I is the number of iterations before convergence. KFDS basically implies a Gram matrix computation and a matrix inversion at each iteration, hence a $O(IDN^3))$ complexity.

The scaled kernel based approaches have been thoroughly compared to the R2W2 method, that also rely on scale factors. We have stated that the R2W2 method requires a double optimization loop. Each iteration of the outer loop involves

1) SVM training to evaluate the α_i factors
2) Evaluation of the full Gram matrix $\boldsymbol{K}$
3) Quadratic programming optimization to evaluate R
4) Computation of the normal vector norm $||\boldsymbol{w_h}||^2$
5) Computation of feature-wise derivative matrices $\partial_d \boldsymbol{K}$

SAS and SCSS only involve steps 2 and 5, plus the necessary additions to compute Frobenius products. They do not need SVM or any quadratic programming solving, which can be costly. In particular, the optimization in step 3 is very costly, and involves the full expression of the Gram matrix in memory, which is not the case for SAS and SCSS. Similarly, most MKL-based methods involve both the Gram matrix computation and SVM solvings. Computational time provided in Section V will confirm that the proposed methods perform faster than R2W2.

C. Computational time comparison

Computational times are compared here between our methods and AROM, R2W2 and RFE. Since SAS and SCSS share most of their process, they are implemented in a common function and therefore have identical speed. SAS and KFDS are entirely written in Matlab, whereas the other all involve the same implementation of SVM in C (Thorsten Joachims' *SVMlight* [39]). Computations were done on the single core of an iMac with 2.66 GHz Intel Core 2 Duo and 4 Go RAM. All iterative algorithms are fixed to 10 iterations. Table VIII shows the evolution of the CPU-time when the number of examples N increases (at fixed dimension $D = 30$), while in Table IX the number of features D increases (with $N = 30$). All durations are normalized by the minimal duration.

N	**AROM**	**R2W2**	**RFE**	**1)SAS**	**3)KFDS**
5	6	14	65	42	2
10	7	17	55	29	1
20	8	20	63	34	1
50	14	32	68	41	2
200	23	292	127	67	3
500	38	2710	353	270	23
1000	88	17764	1021	1153	169

TABLE VIII
COMPARED CPU-TIME OVER N (NORMALIZED BY THE LOWEST TIME).

D	**AROM**	**R2W2**	**RFE**	**1)SAS**	**3)KFDS**
5	78	83	28	110	1
20	83	100	98	150	3
50	88	138	265	208	4
200	105	260	1503	455	25
500	133	498	5715	880	248
1000	153	890	19388	1600	1801
2000	210	1748	80083	3143	14037

TABLE IX
COMPARED CPU-TIME OVER D (NORMALIZED BY THE LOWEST TIME).

Table VIII clearly shows that R2W2 has a much higher complexity that SAS and KFDS when dealing with a large number of examples N. Both SAS and R2W2 are quadratic with N, but sub-linear with the number of features D (Table IX), because once the kernel products are computed, they are independent of the dimension. R2W2 is faster that SAS, when D increases, with a small number of examples N. RFE is quadratic with N but remains acceptable in Table VIII, when compared to other methods. However, its faces a combinatorial explosion when D increases (Table IX).

We now focus on the comparison between KFDS and AROM because their design is similar. KFDS is much faster when D and N remain low, but the cost increases very quickly with high dimensions D. However, AROM is the method with the less Matlab code (only scale factor updates) and relies mostly on the optimized implementation of SVMlight, which might explain the better stability in terms of cost.

VII. CONCLUSION

We have provided here reliable alternatives for state-of-the-art feature selection methods adapted to Support Vector Machines. The Kernel Target Alignment and Kernel Class Separability had not yet been fully explored in the field of feature selection, or only under simplified forms. The methods proposed here, based on a scaled kernel optimization, prove very efficient and comparable in performance to recent SVM-based methods, while being less complex. Comparative studies with former works on these criteria show that the proposed methods perform better. Moreover, the simplicity of the criteria allows for an efficient iterative computation that can scale up to arbitrarily large training sets. Despite its theoretical reliability, KCS is less efficient than KTA, mostly because the regularization attempt to prevent trivial convergence is not sufficient. The Kernel Fisher Discriminant Selection shows comparable results with AROM, at a reduced computational cost, for reasonable amounts of data. However, its use is still limited to the linear kernel, but it sometimes provides surprisingly good results when followed by a non-linear SVM classification. The extension of this method to more complex kernels will be explored in the future. In addition, theory and experiments also show that the SAS and SCSS methods are directly applicable to multiclass problems.

An interesting perspective arises from the strong relationship between the Support Vector Machines and the Fisher Discriminant Analysis. Either in the input space or in the feature space, both methods consist in finding an optimal hyperplane separating the classes' examples. The difference lies in the choice of the optimality criterion. In [43], Shashua proves that the hyperplane of a Support Vector Machine is equivalent to the one found by Fisher Linear Discriminant on the set of its Support Vectors. He then claims that SVMs can be seen as a way to sparsify FLD, thus improving its generalization. Future works will explore the track of restricting the training set of the KCS-based methods to the support vectors identified through an SVM training, in order to reduce complexity and also discard useless or irrelevant information.

REFERENCES

[1] J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the theory of neural computation*. Redwood City, California: Addison-Wesley, 1991.

[2] G. H. John, R. Kohavi, and K. Pfleger, "Irrelevant features and the subset selection problem," in *Proc. Int. Conf. on Machine Learning (ICML '94)*, 1994, pp. 121–129.

[3] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, March 2003.

[4] B. Boser, I. Guyon, and V. Vapnik, "A training algorithm for optimal margin classifiers," *Annual Workshop on Computational Learning Theory*, pp. 144–152, 1992.

[5] V. Vapnik, *Statistical Learning Theory*. Wiley Interscience, 1998.

[6] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik, "Feature selection for SVMs," in *Advances in Neural Information Processing Systems*, vol. 13, 2000, pp. 668–674.

[7] J. Neumann, C. Schörr, and G. Steidl, "Combined SVM-based feature selection and classification," *Machine Learning*, vol. 61, pp. 129–150, November 2005.

[8] L. Wang, "Feature selection with kernel class separability," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30 (9), pp. 1534–1546, September 2008.

[9] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford: Oxford University Press, 1995.

[10] M. Zaffalon and M. Hutter, "Robust feature selection by mutual information distributions," in *Proc. Int. Conf. on Uncertainty in Artificial Intelligence (UAI-2002)*, San Francisco, CA., 2002, pp. 577–584.

[11] K. Kira and L. A. Rendell, "A practical approach to feature selection," in *Proc. Int. Conf. on Machine Learning ML '92*, Aberdeen, Scotland, United Kingdom, 1992, pp. 249–256.

[12] X. He, D. Cai, and P. Niyogi, "Laplacian score for feature selection," in *Advances in Neural Information Processing Systems (NIPS)*. MIT Press, 2006, pp. 507–514.

[13] P. S. Bradley and O. L. Mangasarian, "Feature selection via mathematical programming," *INFORMS Journal on Computing*, vol. 10 (2), pp. 209–217, February 1998.

[14] ——, "Feature selection via concave minimization and support vector machines," in *Proc. Int. Conf. on Machine Learning (ICML '98)*, 1998, pp. 82–90.

[15] J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping, "Use of the zero-norm with linear models and kernel methods," *Journal of Machine Learning Research*, vol. 3, pp. 1439–1491, March 2003.

[16] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Machine Learning*, vol. 46 (1-3), pp. 389–422, 2002.

[17] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing multiple parameters for support vector machines," *Machine Learning*, vol. 46(1-3), pp. 131–159, 2002.

[18] G. R. Lanckriet, N. Cristianini, P. Bartlett, L. ElGhaoui, and M. I. Jordan, "Learning the kernel matrix with semidefinite programming," *Journal of Machine Learning Research*, vol. 5, pp. 27–72, April 2004.

[19] S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf, "Large scale multiple kernel learning," *Journal of Machine Learning Research*, vol. 7, July 2006.

[20] A. Rakotomamonjy, F. R. Bach, S. Canu, and Y. Grandvalet, "SimpleMKL," *Journal of Machine Learning Research*, vol. 9, pp. 2491–2521, November 2008.

[21] Z. Xu, R. Jin, J. Ye, M. R. Lyu, and I. King, "Non-monotonic feature selection," in *Proc. Int. Conf. on Machine Learning (ICML '09)*, Montreal, Canada, 2009, pp. 1145–1152.

[22] M. Tan, L. Wang, and I. W. Tang, "Learning sparse svm for feature selection on very high dimensional datasets," in *Proc. Int. Conf. on Machine Learning (ICML '10)*, haifa, Israel, 2010, pp. 1047–1054.

[23] M. Varma and B. R. Babu, "More generality in efficient multiple kernel learning," in *Proc. of the International Conference on Machine Learning*, Montreal, Canada, June 2009, pp. 1065–1072.

[24] K.-Q. Shen, C.-J. Ong, X.-P. Li, and E. P. V. Wilder-Smith, "Novel multiclass feature selection methods using sensitivity analysis of posterior probabilities," in *IEEE International Conference on Systems, Man and Cybernetics*, Singapore, 12-15 October 2008, pp. 1116–1121.

[25] B. Cao, D. Shen, J.-T. Sun, Q. Yang, and Z. Chen, "Feature selection in a kernel space," in *Proc. Int. Conf. on Machine Learning (ICML '07)*, vol. 227, 2007, pp. 121–128.

[26] L. Song, A. J. Smola, A. Gretton, K. M. Borgwardt, and J. Bedo, "Supervised feature selection via dependence estimation," in *Proc. Int. Conf. on Machine Learning (ICML '07)*. ACM, 2007, pp. 823–830.

[27] N. Cristianini, J. Kandola, A. Elisseeff, and J. Shawe-Taylor, "On kernel target alignment," *Journal of Machine Learning Research*, vol. 1, 2002.

[28] V. Gomez-Verdejo, M. Martinez-Ramon, J. Arenas-Garcia, M. Lazaro-Gredilla, and H. Molina-Bulla, "Support vector machines with constraints for sparsity in the primal parameters," *IEEE Transactions on Neural Networks*, vol. 22, no. 8, pp. 1269–1283, 2011.

[29] S. W. Lee and Z. Bien, "Representation of a fisher criterion function in a kernel feature space," *IEEE Transactions on Neural Networks*, vol. 21, no. 2, pp. 333–339, 2009.

[30] H. Xue, S. Chen, and Q. Yang, "Structural regularized support vector machine: A framework for structural large margin classifier," *IEEE Transactions on Neural Networks*, vol. 22, no. 4, pp. 573–587, 2011.

[31] J. Kandola, J. Shawe-Taylor, and N. Cristianini, "On the extensions of kernel alignment," Department of Computer Science, University of London, Tech. Rep., August 2002.

[32] K.-P. Wu and S.-D. Wang, "Choosing the kernel parameters of support vector machines according to the inter-cluster distance," in *Proc. of the Int. Joint Conf. on Neural Networks*, July 16-21 2006, pp. 1205–1211.

[33] J.-B. Pothin and C. Richard, "A greedy algorithm for optimizing the kernel alignment and the performance of kernel machines," in *Proceedings of EUSIPCO '06*, September 4-8 2006.

[34] R. Vert, "Designing a m-svm kernel for protein secondary structure prediction," Master's thesis, University of Nancy 1 LORIA, 2002.

[35] A. Webb, *Statistical Pattern Recognition*. Wiley, October 15 2002.

[36] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.-R. Müller, "Fisher discriminant analysis with kernels," in *IEEE Neural Networks for Signal Processing IX*, August 1999, pp. 41–48.

[37] H. Xiong, M. N. S. Swamy, and M. O. Ahmad, "Optimizing the kernel in the empirical feature space," *IEEE Transactions on Neural Networks*, vol. 16 (2), pp. 460–474, March 2005.

[38] A. Asuncion and D. Newman, "UCI machine learning repository," 2007.

[39] T. Joachims, *Making Large-Scale SVM Learning Practical*. MIT Press, Cambridge, MA, 1999, pp. 169–184.

[40] G. Richard, M. Ramona, and S. Essid, "Combined supervised and unsupervised approaches for automatic segmentation of radiophonic audio streams," in *Proceedings of ICASSP '07*, April 15-20 2007, pp. 461–464.

[41] T. Hastie and R. Tibshirani, “Classification by pairwise coupling,” in *Advances in Neural Information Processing Systems*, M. I. Jordan, M. J. Kearns, and S. A. Solla, Eds., vol. 10. The MIT Press, 1998.
[42] P. D. Tao and L. T. H. An, “A D.C. optimization algorithm for solving the trust-region subproblem,” *SIAM Journal on Optimization*, vol. 8 (2), pp. 476–505, 1998.
[43] A. Shashua, “On the relationship between the support vector machine for classification and sparsified fisher’s linear discriminant,” *Neural Processing Letters*, vol. 9 (2), pp. 129–139, April 1999.

Mathieu Ramona was bord on June, 19th, 1891, near Paris, France. He received the State Engineering degree from École Normale Supérieure de Techniques Avancées (ENSTA), Paris, France in 2008, and the Ph.D. degree, in the field of audio signal processing, from TELECOM ParisTech (formerly ENST), Paris, France, in 2010. He is currently a postdoctoral scientist in the Sound Analysis/Synthesis Team at the IRCAM.

His main research interests are machine learning for audio indexing, feature selection, musical information retrieval, and audio identification through fingerprinting.

Gaël Richard (SM’06) received the State Engineering degree from TELECOM ParisTech, France (formerly ENST) in 1990, the Ph.D. degree from LIMSI-CNRS, University of Paris-XI, in 1994 in speech synthesis, and the Habilitation à Diriger des Recherches degree from the University of Paris XI in September 2001.

After the Ph.D. degree, he spent two years at the CAIP Center, Rutgers University, Piscataway, NJ, in the Speech Processing Group of Prof. J. Flanagan, where he explored innovative approaches for speech production. From 1997 to 2001, he successively worked for Matra, Bois d’Arcy, France, and for Philips, Montrouge, France. In particular, he was the Project Manager of several large scale European projects in the field of audio and multimodal signal processing. In September 2001, he joined the Department of Signal and Image Processing, Telecom ParisTech, where he is now a Full Professor in audio signal processing and Head of the Audio, Acoustics, and Waves research group. He is a coauthor of over 80 papers and inventor in a number of patents. He is also one of the experts of the European commission in the field of speech and audio signal processing.

Prof. Richard is a senior member of IEEE and Associate Editor of the IEEE Transactions on Audio, Speech and Language Processing.

Bertrand David (M’06) was born on March 12, 1967 in Paris, France. He received the M.Sc. degree from the University of Paris-Sud, in 1991, and the Agrégation, a competitive French examination for the recruitment of teachers, in the field of applied physics, from the École Normale Supérieure (ENS), Cachan. He received the Ph.D. degree from the University of Paris 6 in 1999, in the fields of musical acoustics and signal processing of musical signals. He formerly taught in a graduate school in electrical engineering, computer science and communication. He also carried out industrial projects aiming at embarking a low complexity sound synthesizer. Since September 2001, he has worked as an Associate Professor with the Signal and Image Processing Department, TELECOM ParisTech (formerly ENST). His research interests include parametric methods for the analysis/synthesis of musical and mechanical signals, spectral parametrization and factorization, music information retrieval, and musical acoustics.