Interpretable support vector machines for functional data

Interpretable Support Vector Machines for Functional Data

Belen Martin-Barragan, Rosa Lillo, Juan Romo

Department of Statistics
Universidad Carlos III de Madrid (Spain)

Abstract

Support Vector Machines (SVM) has been shown to be a powerful nonparametric classification technique even for high-dimensional data. Although predictive ability is important, obtaining an easy-to-interpret classifier is also crucial in many applications. Linear SVM provides a classifier based on a linear score. In the case of functional data, the coefficient function that defines such linear score usually has many irregular oscillations, making it difficult to interpret.

This paper presents a new method, called *Interpretable Support Vector Machines for Functional Data*, that provides an interpretable classifier with high predictive power. Interpretability might be understood in different ways. The proposed method is flexible enough to cope with different notions of interpretability chosen by the user, so the obtained coefficient function can be sparse, linear-wise, smooth, etc. The usefulness of the proposed method is shown in real applications getting interpretable classifiers with comparable, sometimes better, predictive ability versus classical SVM.

Keywords: Data mining, interpretability, classification, linear programming, regularization methods, functional data analysis

1. Introduction

Roughly speaking, the objects of study in Functional Data Aanalysis (FDA) are functions. As functions we understand curves, surfaces or anything else varying over a continuum. Although the continuum is often time, it might also be other things: location, wavelength, probability, etc. Concrete values of this continuum are sometimes referred to as time points in order to make the description more intuitive.

We deal with the problem of classifying functional data. Suppose we observe a binary response Y (the class) to a functional predictor X, where $X \in \mathcal{X}$ is a function defined on the bounded interval $\mathcal{I}$, i.e. $X : \mathcal{I} \mapsto I\!R$, and $\mathcal{X}$ is given set

Email addresses: belen.martin@uc3m.es (Belen Martin-Barragan), lillo@est-econ.uc3m.es (Rosa Lillo), juan.romo@uc3m.es (Juan Romo)

of functions. Our aim is to construct a classification rule that predicts Y to a given functional datum X with good prediction ability and some interpretability properties.

The classification rule is based on the sign of the so-called *score function* f. The score function is an operator $f : \mathcal{X} \mapsto I\!R$ that, for a given function X, assigns a real number. Since our aim is interpretability, we consider the score function to be a linear operator $T_{\beta,\omega}$ with coefficient function $w \in \mathcal{X}$ and intercept $\beta \in I\!R$:

$$f(X) = T_{\beta,w}X = \int_{\mathcal{I}} w(t)X(t)dt + \beta = \langle w, X\rangle + \beta, \tag{1}$$

where $\langle f, g\rangle = \int_{\mathcal{I}} f(t)g(t)dt$. The estimation of the coefficient function w on the whole interval $\mathcal{I}$ is an infinite dimensional problem. This issue is addressed via regularization, which simultaneously allows as to address our other concern: interpretability.

As in standard Support Vector Machines (SVM), $w(t)$ determines the discriminative power of $X(t)$. For example, areas where $w(t)$ is zero or small has none or low discrimination power, whereas for $|w(t)|$ large, one can expect the behavior of $X(t)$ to influence the classification. This idea provides a clear interpretation of $w(t)$ at a particular time point t, but getting a general idea about the coefficient function w requires it to be simple: for example, if $w(t)$ has unnatural wiggles all along the interval $\mathcal{I}$, it would be difficult to interpret its behavior.

In different applications the simplicity of w might be understood in different ways. For instance, a coefficient function that is non-zero in just a few points, could detect the few points that are more relevant in classification. This idea has been proposed within a logistic regression model, see Lindquist & McKeague (2009). In other situations, one might prefer a coefficient function that is constant over a few subintervals of $\mathcal{I}$ and zero on the rest. A method that detects a few segments with high discriminative power have been proposed in Li & Yu (2008) by combining feature selection, classical linear discriminant analysis and SVM. In gene expression analysis, detection of relevant segments are also quite desirable because relevant genes are expected to be located close to each other along the chromosome (Rapaport et al., 2008). All this literature propose different methodologies for different notions of interpretability. Our proposal deals with all these notions under a common framework.

We borrow the interpretability notions proposed by James et al. (2009) for functional linear regression. The idea is to enforce one or several derivatives of the coefficient function w to be sparse. Which derivatives are enforced to be sparse depends on the notion of interpretability preferred by the practitioner. This paper proposes a new method, which we call *Interpretable Support Vector Machines for Functional Data* (ISVMFD) that produces SVM-based classifiers for functional data which have high classification accuracy and whose coefficient function are easy to interpret. The problem is formulated as a linear program, in the framework of L_1-norm SVM.

The outline of the paper is as follows: Section 2 reviews classic and recent literature on the main tools related to our method: FDA and SVM. Recent efforts to obtain interpretable SVM-based classifiers for multivariate data are also reviewed in Section 2. In Section 3 the ISVMFD method is introduced and it is proposed to implement it through the use of a basis. Section 4 studies how other methods available in the literature are particular cases of ISVMFD. A wide study with real-world datasets is presented in Section 5 and finally, in Section 6, several conclusions are driven.

2. Literature review

The term Functional Data Analysis was first used in Ramsay & Dalzell (1991) two decades ago. Since them, especially in the last decade, it has become a fruitful field in statistic. The range of real world applications where the objects can be thought as functions is as diverse as speech recognition, spectometric, metheorology or clients segmentation to cite just a few (Algirdas & Laukaitis, 2008; Ferraty & Vieu, 2003; James et al., 2009; Laukaitis & Rackauskas, 2005). A good review of the different FDA techniques applied to real world problems can be found in Ramsay & Silverman (2002). For a deeper insight into the subject see e.g. Ferraty & Vieu (2006) and Ramsay & Silverman (2005).

In spite of its continuous nature, the functions under study are usually collected in a discrete manner. Hence, every function is represented by a high-dimensional vector with highly correlated coordinates. Direct use of these multivariate techniques to functional data is possible, but often works bad in practice. FDA makes use of the functional nature of the data to extend such techniques to their functional counterparts. This way, the different classical multivariate techniques have been extended to functional data. Principal functional component analysis (Dauxois et al., 1982) was pioneer among these techniques. Functional regression has also been widely studied, both in the case in which the response is also functional (see e.g. Cuevas et al. (2002); Faraway (1997); Liang & Zeger (1986)) , and the case in which the response is scalar (see e.g. Baíllo & Grané (2009); Cardot & Sarda (2005); James (2002)). Classification or discriminant analysis has also been taken into account (Ferraty & Vieu, 2003; James & Hastie, 2001; Leng & Müller, 2005). Other interesting approaches include, for instance, a new concept of depth based on the band formed by two functions (Lopez-Pintado & Romo, 2009).

We focus in this paper on the binary supervised classification problem, where two classes $\{-1, 1\}$ of curves need to be discriminated. SVM (Cortes & Vapnik, 1995; Moguerza & Muñoz, 2006; Vapnik, 1995) have become very popular during the last decade. The basic idea behind SVM can be explained geometrically. If we think in the data as living in a $p-$dimensional space, SVM finds the separating hyperplane with maximal margin, i.e., the one furthest away from the closest object. This geometrical problem is expressed as a smooth convex problem with linear constraints, solved either in its primal or dual form. Another interpretation can be done in terms of the regularization theory where the hinge loss plus a quadratic regularization penalty is minimized (Hastie et al., 2001;

Tibshirani, 1996). The most popular and powerful versions of SVM embed the original variables into a higher dimensional space (Herbrich, 2002). This embedding is usually implicitly specified by the choice of a function called kernel.

Extensions of SVM to functional data have been proposed in Muñoz & González (2010) and Rossi & Villa (2006). In Muñoz & González (2010), SVM is used to represent the functional data by projecting the original functions onto the eigenfunctions of a Mercer Kernel. Rossi & Villa (2006) definine new classes of kernels that take into account the functional nature of the data. Two types of functional kernels are proposed: projection-based kernels and transformation-based kernels. In projection-based kernels, the idea is to reduce the dimensionality of the input space, i.e. to apply the standard filtering approach of FDA. Transformation-based kernels allow to take into account expert knowledge (such as the fact that the curvatures of a function can be more discriminant than its values in some applications).

With multivariate data, kernels provides an implicit way to get a nonlinear classifier, by projecting the data in the higher dimensional space induced by the kernel. The final classifier is nonlinear in the original space, but linear in the projected space. Functional data are already high dimensional and the high dimensionality is usually the cause of problems, hence the use of kernels to project data in a higher dimensional space seem to be less crucial. Moreover, the kernel-based classifier would be easy to interpret in the projected space, but not in the original. We focus on the linear kernel in our method.

The interpretability issue in SVM has already been addressed for multivariate data. The first attempts to make SVM more interpretable consist on a two-step procedure: first, SVM is run, and then a rule, resembling the SVM-classifier but easier to interpret, is built. See e.g. Baesens et al. (2003); Barakat & Diederich (2006); Martens et al. (2007, 2009). One obtains an alternative classifier which hopefully get similar predictions, but is more interpretable. Recently, a two-stage iterated method is proposed for credit decision making (Li et al., 2011), which combines feature selection and multi-criteria programming. In Carrizosa et al. (2010, 2011), one-step SVM-based procedures are proposed to get the relevant variables and the relevant interactions between variables. Although one would expect classification rates to be deteriorated when looking for interpretable classifiers, the experiments in Carrizosa et al. (2010, 2011) show that their proposals are competitive with SVM. See Baesens et al. (2009); Lessmann & Voß (2009); Van Gestel et al. (2007); Verbeke et al. (20011) for other recent references on the topic.

3. Methodology

3.1. Interpretable Support Vector Machines for Functional Data

Let $\{X_u, Y_u\}_{u=1}^n$ be a sample of n functional data $X_u \in \mathcal{X}$ together with its class $Y_u \in \{-1, 1\}$. The classical SVM with the linear kernel seeks for the

coefficient function w that minimizes

$$min_{w,\beta}\|w\|_p^p + C\sum_{u=1}^{n} h\left(y_u,\, \langle w, X_u\rangle + \beta\right) \tag{2}$$

where $\|\cdot\|_p$ is the $p-$norm, $h(y,s) = (1-ys)_+$ is the hinge loss and C is a tuning parameter that trades off the regularization term $\|\omega\|_p^p$ and the loss term.

The class is predicted as the sign of the score function given in (1). In case of ties, i.e. $f(X) = 0$, prediction can be randomly assigned or following some predefined order. Throughout this article, following a worst case approach, ties will be considered as misclassifications.

Although the regularization with the Euclidean norm is the most common, other norms have also been applied. For instance, the L_1 norm is known to be good when a sparse coefficient vector is desirable. Bradley & Mangasarian (1998) demonstrated the usefulness of penalties based on the L_1 norm in classification problems. In regression, LASSO (Tibshirani, 1996) and the Dantzig selector (Candes & Tao, 2007) also successfully use the L_1 norm in high-dimensional problems.

In order to get the interpretable classifier, we propose a modified version of SVM that we call Interpretable Support Vector Machines for Functional Data (ISVMFD). Following the concepts of interpretability described in Section 1, we propose to use a different regularization term that depends on the preferences of the user for the interpretability notion. The user must select one or several derivatives to be sparse. For example, if the user is concerned with detecting relevant time points, the zero derivative (the actual w) is selected to be sparse. Sparsity of the first derivative leads to constant-wise w which is useful to identify relevant segments. A user might prefer a coefficient function that is zero over large regions, but smooth quadratic-wise where it is nonzero. In this case, sparsity on both the zero and the third derivative is sought.

Let $\mathcal{D}$ be the set of the chosen derivatives. The proposed regularization term is $\sum_{d\in\mathcal{D}}\|w^{(d)}\|_1$, where $\|\cdot\|_1$ is the L_1 norm and $w^{(d)}$ is the $d-$th derivative of w or an approximation of it. This yields to the following optimization problem,

$$min_{w\in\mathcal{X},\beta\in I\!R}\sum_{d\in\mathcal{D}}\|w^{(d)}\|_1 + C\sum_{u=1}^{n} h(y_u, < w, X_u > +\beta). \tag{3}$$

Note that when several derivatives are included in $\mathcal{D}$, it might also be convenient to give different weights to the different derivatives. We do not explore such issue, but it is a straightforward modification of (3).

The set of functions $\mathcal{X}$ can be a wide space, such as L^2, for which Problem (3) become infinite dimensional. This issue is addressed in the next section via the use of a basis.

3.2. Implementation through the use of a basis

We consider the selection of a $p-$dimensional basis $B(t) = [b_1(t), b_2(t), \ldots, b_p(t)]^\top$, in such way that:

$$w(t) = B(t)^\top\eta. \tag{4}$$

Usually, p is assumed to be low in order to provide some form of regularization that avoids overfitting. However we work with p large enough to allow a perfect fitting. In our method, regularization is not based on the low dimension of B, but it is intrinsically related to the interpretability issue, as it is done by minimizing the L_1 norm of one or several derivatives of the score function w.

Our method can be applied to any high dimensional basis, such as Fourier, splines or wavelets. To keep it simple, one might think on a simple grid basis,

$$b_i(t) = \begin{cases} 1 & \text{if } t \in [t_{i-1}, t_i] \\ 0 & \text{otherwise,} \end{cases} \tag{5}$$

for all $i = 1, \ldots, p$.

Once we have a basis B, the score function can be rephrased as:

$$f(X_u) = \eta^\top x_u + \beta, \tag{6}$$

where $x_u = \int_{\mathcal{I}} X_u(t) B(t) dt$.

In principle, we do not assume the basis functions $B(t)$ to be differentiable. That is the case, for instance, of the base function proposed in (5). Based on the choices of the practitioner, we are seeking a score function w that is sparse, constant-wise, linear-wise, quadratic-wise, etc. We propose to approximate the derivatives of $w(t)$ by its finite differences. Let $s_0, s_1, \ldots, s_r$ be a fine grid of the interval $\mathcal{I}$. This grid does not necessarily coincide with the grid used in (5), although this is the option used in our numerical experiments. Denote $D^0 w = (w(s_0), w(s_1), \ldots, w(s_r))^\top$ the discretization of the coefficient function w on such grid. An approximation of the $d-$th derivative of w can be obtained by the finite difference operator, that is

$$D^d w(s_j) \quad = \frac{D^{d-1} w(s_j) - D^{d-1} w(s_{j-1})}{s_j - s_{j-1}} \quad \text{for } j = 0, 1, \ldots, r-d. \tag{7}$$

Enforcing sparsity on $D^d w = (D^d w(s_0), D^d w(s_1), \ldots, D^d w(s_{r-d}))^\top$ yields a coefficient function w whose $d-th$ derivative is zero in all but a few points s.

Let $A_d = [D^d B(s_0), D^d B(s_1), D^d B(s_2), \ldots, D^d B(s_{r-d})]^\top$, where D^d is the finite difference operator defined in (7). Then, $\gamma = A_d \eta = D^d w$ is a good approximation of $w^{(d)}$ and hence, enforcing sparsity in γ pushes $w^{(d)}$ to be zero at most points t.

With this setting, (3) reduces to the vector optimization problem

$$min_{\eta,\beta} \sum_{d \in \mathcal{D}} \|A_d \eta\|_1 + C \sum_{u=1}^{n} h(y_u, \eta^\top x_u + \beta), \tag{8}$$

which can be rephrased as the linear program:

$$\begin{array}{lll} min & \sum_{d \in \mathcal{D}} e_{r+1-d}^\top z_d + C \sum_{u=1}^n \xi_u & \\ \text{s.t.} & y_u(x_u^\top \eta + \beta) + \xi_u \geq 1, & u = 1, 2, \ldots, n, \\ & -z_d \leq A_d \eta \leq z_d, & d \in \mathcal{D}, \\ & \xi_u \geq 0, & u = 1, 2, \ldots, n, \\ & z_d \in I\!R^{r+1-d}, & d \in \mathcal{D}, \\ & \eta \in I\!R^{p+1}, & \\ & \beta \in I\!R, & \end{array} \tag{9}$$

where e_i is the $i-$dimensional vector with value one at each component.

Take for instance the case of the grid basis defined in (5). Suppose each function X_u is defined on the interval $\mathcal{I} = [0, p]$ and the grid $(0, 1, 2, \ldots, p)$ is considered. It can be easily seen that, $\eta = (w(0), w(1), \ldots, w(p))^\top$, A_0 is the identity matrix and $A_d = A_1^\top A_{d-1}$ for $d = 2, \ldots, p$. For example, A_1 and A_2 are equal to:

$$A_1 = \begin{pmatrix} 1 & -1 & 0 & \ldots & 0 \\ 0 & 1 & -1 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & \ldots & -1 \\ 0 & 0 & 0 & \ldots & 1 \end{pmatrix} \quad \text{and} \quad A_2 = \begin{pmatrix} 1 & -2 & 1 & \ldots & 0 \\ 0 & 1 & -2 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & \ldots & -2 \\ 0 & 0 & 0 & \ldots & 1 \end{pmatrix}. \tag{10}$$

4. ISVMFD as a global framework of several existing methods

In this section we study how ISVMFD can be seen as a generalization of other methods available in the literature. In particular, L_1-norm SVM (Bradley & Mangasarian, 1998; Carrizosa et al., 2010, 2011; Pedroso & Murata, 2001) and Fused SVM (Tibshirani et al., 2005; Rapaport et al., 2008) turn out to be particular cases of ISVMFD for particular choices of the derivatives.

For linear SVM applied to vectors instead of functions, the L_1-norm SVM is a modification of SVM where the quadratic penalty term is replaced by the L_1-norm penalty of the coefficient vector. See for instance Bradley & Mangasarian (1998) and Zhu et al. (2003).

For simplicity in the notation suppose that $\mathcal{I} = [0, 1]$. Let $t_i = i/p$, for $i = 1, 2, \ldots, p$ be a a regular grid on $[0, 1]$. Suppose the functional datum X_u is known only on such grid. Consider that ISVMFD is used to select several time points. This means that the set of derivatives $\mathcal{D}$ in (3) should be set to $\{0\}$. We can represent the coefficient function w using a grid basis as in (5). Since X_u is unknown in the open interval (t_{i-1}, t_i), we consider $\frac{1}{p}X(t_i)$ as an approximation of

$$\int_{\mathcal{I}} X(t)b_i(t)dt = \int_{t_{i-1}}^{t_i} X(t)dt.$$

With this setting, application of ISVMFD to functions $\{X_u\}_{u=1}^n$ reduces to solving (8) with

$$x_u = \frac{1}{p}\left(X_u(t_1), X_u(t_2), \ldots, X_u(t_p)\right)^\top, \quad \text{for all } u = 1, 2, \ldots, n.$$

In L_1-norm SVM, the L_1-norm penalty is known to act as a feature selection problem because it enforces the coefficient vector to be sparse. Hence, the L_1-norm SVM is able to produce a classifier that detects the several time points

that are more relevant for classification. L_1-norm SVM applied directly to the vectors $\{x_u\}_{u=1}^n$ reduces to solving the following problem:

$$min_{\omega \in I\!R^p, \beta \in I\!R} \|\omega\|_1 + C \sum_{u=1}^{n} h(y_u, \omega^\top \hat{x}_u + \beta), \tag{11}$$

which is equivalent to (8).

Another method that can be seen as a particular case of ISVMFD is the Fused SVM. Fused SVM is the SVM-based counterpart of Fused Lasso, both proposed in Tibshirani et al. (2005). Fused Lasso is a generalization of Lasso designed for problems whose features can be ordered in some meaningful way. It encourages both sparsity of the coefficient vector and sparsity of the differences between two consecutive components of the coefficient vector. Fused SVM seeks for a coefficient vector w that optimizes the following linear program:

$$\begin{array}{lll} min & \sum_{u=1}^n \xi_u & \\ \text{s.t.} & y_u(x_u^\top \eta + \beta) + \xi_u \geq 1, & u = 1, 2, \ldots, n, \\ & \sum_{j=1}^p |w_j| \leq s_1, & \\ & \sum_{j=2}^p |w_j - w_{j-1}| \leq s_2, & \\ & \xi_u \geq 0, & u = 1, 2, \ldots, n, \\ & w \in I\!R^p, & \\ & \beta \in I\!R, & \end{array} \tag{12}$$

where s_1 and s_2 are two tuning parameters that trade off the loss term and the regularization terms (sparsity of w and sparsity of the differences). This problem is known to be equivalent to

$$\begin{array}{lll} min & \sum_{u=1}^n \xi_u + \lambda_1 \sum_{j=1}^p |w_j| + \lambda_2 \sum_{j=2}^p |w_j - w_{j-1}| & \\ \text{s.t.} & y_u(x_u^\top \eta + \beta) + \xi_u \geq 1, & u = 1, 2, \ldots, n, \\ & \xi_u \geq 0, & u = 1, 2, \ldots, n, \\ & w \in I\!R^p, & \\ & \beta \in I\!R, & \end{array} \tag{13}$$

in the sense that, for any positive s_1 and s_2 on (12) there exist $\lambda_1, \lambda_2 > 0$, such that (η, β, ξ) is optimal for (12) if and only if it is optimal for (13).

Taking $\lambda_1 = \lambda_2$ and $C = \frac{1}{\lambda_1}$, (13) is the problem obtained when applying ISVMFD with $\mathcal{D} = \{0, 1\}$ and the grid basis (5).

5. Illustration on real databases

5.1. Spectometric data

The Tecator[1] data set consists of 215 near-infrared absorbance spectra of meat samples. These data are recorded on a Tecator Infratec Food and Feed

[1]The data set is available at http://lib.stat.cmu.edu/datasets/tecator

Analyzer working in the wavelength range 850 - 1050 nm by the Near Infrared Transmission (NIT) principle. Each sample contains finely chopped pure meat with different moisture, fat and protein contents. For each meat sample the data consists of a 100 channel spectrum of absorbances and the contents of moisture (water), fat and protein. The absorbance is -log10 of the transmittance measured by the spectrometer. The three contents, measured in percent, are determined by analytic chemistry.

Figure 1 shows the spectra of the samples with high (left) and low (right) fat contents. The most important difference betweeen these two sets of curves seems to be in their shape. High-fat curves tends to have two local minima whereas low-fat have only one. This suggests, as pointed out previously in Rossi & Villa (2006), to use the second derivative of the these curves instead of the original curves. Figure 2 shows the curvature (second differences) of the curves.

For fair comparison with their results, we follow the same experimental setting as in Rossi & Villa (2006). Hence, we focus in the discrimination of samples with a low-fat content (less than 20%) versus high fat content (more than 20%). The dataset is split into 120 spectra for learning and 95 for testing. This splitting is repeated 250 times. For each splitting, the training set is again divided in two subsets: 60 spectra for learning and 60 spectra for validation. For each training set, the SVM is run in the learning set with the trade-off parameter of SVM, C, set to 10^i for $i = -1, 1, \ldots, 8$. The C with the best performance in the validation set is chosen and the SVM with such C is run again in the training set. Finally, the obtained classifier is evaluated in the testing set. This process is repeated 250 times and the average error on the testing set over the 250 repetitions is given. In all the experiments we use CPLEX 12.1 to solve the linear program (9). The whole algorithm is programmed in Matlab and it is available under request.

As suggested in Figures 1 and 2, and in the empirical results obtained in Rossi & Villa (2006) and Li & Yu (2008), the second derivative of the spectra is more discriminative than the spectra itself. Hence, we focus on the use of such a second spectra. To approximate the second derivative, Rossi & Villa (2006) uses a fixed spline subspace to represent the functions so as to calculate the seconde derivative. Instead of that, we apply the second finite diference operator D^2 defined in (7) to each function X_u. Classical linear SVM applied to this transformed data yields an error of 1.8779%, which is better than the results reported in Rossi & Villa (2006) for FSVM (3.28% for the linear kernel and 2.6% for the Gaussian kernel). This example is also used in González & Muñoz (2010) where each functional datum is projected onto a Reproducing Kernel Hilbert Space (RKHS). Different kernels and different classifiers are tried. Among them, the best classification error reported is 1.54%.

In each practical application, the interpretability of the coefficient function issue might mean something different. For example, some practitioners might prefer to get a very sparse coefficient function, whereas others might prefer a linear-wise one. Different choices for the set of derivatives $\mathcal{D}$ yield different interpretation effects for the coefficient function. We have tried several sensible choices for these derivatives in order to compare them. Table 1 provides the

interpretation effect and the classification error. The coefficient functions obtained for the first 10 runs are depicted in Figures 4 and 5 (left), the first of them is depicted on the right size to improve visualization.

$\mathcal{D}$	interpretation effect	error
0	sparse	1.0821**
0 and 1	sparse and constant-wise	1.2800*
0 and 2	sparse and linear-wise	1.2968*
0 and 3	sparse and quadratic-wise	1.3558*
1	constant-wise	1.5368*
2	linear-wise	1.8232
3	quadratic-wise	2.1600
linear FSVM	none	3.28
Gaussian FSVM	none	2.6
linear SVM	none	1.8779
FSDA	detection of segments	1.09
RKHS	none	1.54

Table 1: Classification accuracy in `tecator` database. ** Significantly better (ttest) than all the others; * significantly better (ttest) than SVM.

The best result in terms of classification performance is obtained for the sparse coefficient function. This error is very similar to the one provided in Li & Yu (2008) (1.09 %) by Functional Segment Discriminant Analysis (FSDA), a method that consists in a two-stage feature extraction followed by the application of SVM.

Note that the horizontal axis of Figure 2 represents the wavelength channel where the absorbance is measured. In this application, the detection of the channels with higher discriminative power is a key problem. Figure 4 shows that direct application of ISVMFD clearly detects channel 935 as the most discriminative channel. Figure 6 shows, for every channel, the relative frequency of being selected by ISVMFD over the 250 replications. It is clear that the channel 935 is selected almost always (99.2%), channels around it are also selected quite often and other channels are selected with a frequency bellow 15%. In Li & Yu (2008) a similar experiment is reported for FSDA, with 50 replications, where the channel selected most frequently is also 935, but two other channels 905 ad 1045 are selected at remarkable frequencies too. Classical SVM, apart from getting worse classification ability, cannot be easily used to detect relevant channels as can be seen in Figure 3 where the coefficient vector is shown.

5.2. Phoneme

We consider the phoneme database[2], previously used e.g. in Hastie et al. (2001); Li & Yu (2008) and Rossi & Villa (2006). This dataset is part of TIMIT

[2]The data set is available at http://www-stat.stanford.edu/ tibs/ElemStatLearn/

database and consits on log-periodograms of recorded phonemes of 32 ms duration (the length of each log-periodogram is 256). Following Rossi & Villa (2006) we focus on the phonemes 'aa' versus 'ao'. The curves are shown in Figure 7. We split the dataset in a training sample (with 519 'aa' examples and 759 'ao' examples) and a testing sample (with the rest). The training sample is then divided into a learning and a validation sample (each with the 50% of the training sample) to choose the trade-off parameter C as in Section 5.1. The training/testing split is repeated 20 times.

The classification errors reported in Rossi & Villa (2006) and Li & Yu (2008) are 19.4%, 22% and 18.5% for linear FSVM, rfb FSVM and FSDA respectively. SVM applied to the crude data yields an error of 22.08%. In González & Muñoz (2010) the best classification error reported is 18.14% for RKHS. Table 2 provides the interpretation effect and the classification error for different choices of $\mathcal{D}$. In this example, it seems that all but SVM have comparable classification ability. ISVMFD using a combination of sparsity of w and any other of the subsequent derivatives are slightly better than the other approaches (FSDA, FSVM and ISVMFD with the sparsity effect).

$\mathcal{D}$	interpretation effect	error
0	sparse	19.3052
0 and 1	sparse and constant-wise	17.6879
0 and 2	sparse and linear-wise	17.7790
0 and 3	sparse and quadratic-wise	17.6651
1	constant-wise	18.7244
2	linear-wise	18.5080
3	quadratic-wise	18.2574
linear FSVM	none	19.4
Gaussian FSVM	none	22
linear SVM	none	22.08
FSDA	detection of segments	18.5
RKHS	none	18.14

Table 2: Classification accuracy in `phoneme` database.

The coefficient functions obtained for the first 10 runs are depicted in Figures 9 and 10 (left), the first of them is depicted on the right size to improve visualization. Since data are log-periodograms, the horizontal axis represents the frequency. Take for instance the graphic on the bottom-right corner in Figure 9, we see how there is a region of almost irrelevant frequencies between 100 and 140. In general, the area before 50 seems to be the most relevant for classification. There are several picks there. Around numbers 15 and 30, the coefficient function is negative, what indicates that a high value of the log-periodogram at these frequencies indicates a tendency to be classified in the negative class (the 'ao' phoneme). However, between them there is a region, from channels 23 to 26, where the coefficient is positive, indicating that high values at these num-

bers are representative of the other class. A similar behavior can be observed around numbers 41 and 46. None of these interpretations can be obtained with SVM whose coefficient vector can be found in Figure 8. We can see that no area of the channel spectrum seems more influent than another. The score function randomly oscillates around zero along the whole curve.

5.3. Mitochondrial calcium data set

Biochemical studies suggest that higher levels of mitochondrial calcium overload, a measure of the mitochondrial calcium ion Ca2+ levels, indicate a better protection against the ischemia process. The mitochondrial calcium overload has been monitored in isolated mouse cardiac cells. In each cell, measurements were taken every 10s during one hour (360 time instants). The mitochondrial calcium overload was measured in two groups (control and treatment) with 45 and 44 cells, respectively. We refer to this dataset as the `ca` dataset. In our experiment, we analyze the ability of the curves to discriminate between the treatment and the control group. This dataset has been used in Baíllo et al. (2010).

Since the number of curves is small, we follow a leave-one-out approach, where the training set is formed by all but one curve. This split is repeated for each curve in the dataset. Parameter C is chosen using half the training sample as learning set, and the rest for validation. The SVM classifier directly applied to the data achieves a classification error of 1.236%. The original data can be seen in Figure 11 and the results of ISVMFD for different choices of the interpretation effect can be seen in Table 3. In this case, the classification error is identical for all the interpretation effects that encourage sparsity (zero derivative) and it coincides with the crude SVM error. Figure 13 shows the coefficient functions for such cases.

$\mathcal{D}$	interpretation effect	error
0	sparse	1.1236
0 and 1	sparse and constant-wise	1.1236
0 and 2	sparse and linear-wise	1.1236
0 and 3	sparse and quadratic-wise	1.1236
1	constant-wise	4.4944
2	linear-wise	6.7416
3	quadratic-wise	5.6180
linear SVM	none	1.236
k-NN (uniform metric)*	none	21
k-NN (PLS-based semimetric)*	none	34
nonparametric plug-in*	none	15

Table 3: Classification accuracy in `ca` database. * After elimination of the first part of the data.

Looking at the curves in Figure 11, it seems that the largest differences among the two classes are in the area after the first three minutes. In Baíllo

et al. (2010), this part of the curves is eliminated from the study. It is stated that in many cases the first three minutes each curve shows oscillations which correspond to normal contractions of the cells. This part has high variability and depends on uncontrolled factor. However, in this study we consider the whole set of curves and let the proposed method to show the discriminative power of each part of the curve.

The results reported by Baíllo et al. (2010) are 21%, 34% and 15%, respectively for the k-NN (with uniform metric and PLS-based semimetric) and the nonparametric plug-in discrimination rules. All of these methods were applied after elimination of the high variability part. However, our results using SVM and most versions of ISVMFD in the entire curves give a classification error of 1.1236%.

In this case, ISVMFD does not improve the classification error of SVM, so the advantages are mainly in the interpretability of the results. Figure 12 shows the coefficient function obtained by SVM. The high variability of the curves in the first minutes is reflected in the values of the coefficient function for such minutes. This suggests that the first part of the curves has higher discriminative power. However, the coefficient function makes many wiggles around zero, so it is difficult to see in what intervals this discriminative power is in favor or against the positive class. Let us take a look at Figure 13, where the coefficient functions obtained by ISVMFD is shown. If we are interested only in detecting a few discriminative time points, the curves at the top seems to do a good work. This case, a convenient choice is $\mathcal{D} = \{0\}$, which detects the three most relevant values at the first part of the time interval, and then several others less relevant ones. The two first relevant time points, sorted along the time of occurrence, have an impact in favor of the negative class, whereas the third one has impact in favor of the positive class. Hence, ISVMFD is useful to detect the relevant exact time points and its impact on classification.

The interpretation of the relevance of several *exact* time points might not suit the doctors for medical interpretation. In case they think that the influence of the mitochondrial calcium overload over the class changes smoothly over time, other choices of $\mathcal{D}$ better suits their needs. For example, looking at the bottom of Figure 13, i.e. $\mathcal{D} = \{0, 3\}$, we observe that the score function is zero in large areas of the time interval, and quadratic-wise in the rest. We again see that the most relevant part is in the first part of the curve, but we also observe something more in its behavior. It starts impacting in favor of the negative class, and this impact is increasing until a pick where it starts to decrease until it reaches another pick where the impact in favor of the positive class is the highest, and again this impact decreases until it reaches an area of no impact either in favor of the negative nor the positive class. A slight impact can be observed later on, around the time 210.

The good error rates obtained when using the whole curve, compared with the results in Baíllo et al. (2010), supports the conclusion about the relevance of the first part of the curves. We run also SVM eliminating the first part of the curves and obtained a classification error of 4.49%, that is worse that the results obtained using SVM on the whole curves.

5.4. Weather data

The `weather` dataset consists of one year of daily temperature measurements from each 35 Canadian weather stations. Two experiments are conducted with this data, considering two different classification tasks: `regions` (Atlantic climate vs. the rest) and `rain` (two classes are consider depending if the total yearly amount of precipitations are above or below 600). This experiment is inspired in the good interpretability results obtained in James et al. (2009) for functional regression.

In this experiment, we follow a leave-one-out approach, where the training set is formed by all but one curve. This split is repeated for each curve in the dataset. Parameter C is chosen using half the training sample as learning set, and the rest for validation.

For the classification of `regions`, the original data can be seen in Figure 15 and the results of ISVMFD for different choices of the interpretation effect can be seen in Table 4. In this case, the best classification errors are obtained for $\mathcal{D} = \{0\}$, $\mathcal{D} = \{0, 2\}$ and $\mathcal{D} = \{0, 3\}$, which corresponds to sparse function, combination of sparse together with piece-wise linear function and combination of sparse together with piece-wise quadratic function. The SVM classifier directly applied to the data achieves a classification error of 5.7143%. Figure 17 shows the coefficient functions for different $\mathcal{D}$. For instance, in the case in which sparsity and piece-wise linearity is encouraged, $\mathcal{D} = \{0, 2\}$, we observe two main picks: the first one in February, impacting in favor of the negative class, and the second at the end of November, impacting in favor on the positive class. In contrast, the coefficient function obtained by SVM, shown in Figure 16, is difficult to interpret.

$\mathcal{D}$	interpretation effect	error
0	sparse	2.8571
0 and 1	sparse and constant-wise	5.7143
0 and 2	sparse and linear-wise	2.8571
0 and 3	sparse and quadratic-wise	2.8571
1	constant-wise	8.5714
2	linear-wise	11.4286
3	quadratic-wise	17.1429
SVM	none	5.7143

Table 4: Classification accuracy in `regions` database.

For the classification of `rain`, the original data can be seen in Figure 19 and the results of ISVMFD for different choices of the interpretation effect can be seen in Table 5. In this case, contrary to the situation in previous examples, we face a situation in which encouraging sparsity yields, in general, worse results in terms of error rates. The best result is obtained for $\mathcal{D} = \{2\}$, which encourages piece-wise linearity, without sparsity of the coefficient function itself. In Figure 22, we see how the impact in favor of the positive class increases until mid

March and then decreases until mid September, where it starts to increase again. Again, the interpretation of the coefficient function obtained by SVM, shown in Figure 20, is quite difficult.

$\mathcal{D}$	interpretation effect	error
0	sparse	11.4286
0 and 1	sparse and constant-wise	11.4286
0 and 2	sparse and linear-wise	11.4286
0 and 3	sparse and quadratic-wise	11.4286
1	constant-wise	5.7143
2	linear-wise	2.8571
3	quadratic-wise	5.7143
SVM	none	8.5714

Table 5: Classification accuracy in `rain` database.

6. Conclusions

In this paper we face the problem of obtaining an SVM-based classifier for functional data that has good classification ability and provides a classifier easy to interpret. The interpretability issue might strongly depend on the applications and the preferences of the user. Hence, we consider a flexible framework where different properties of the coefficient function are allowed. ISVMFD generalizes two other proposals available in the literature: the L_1-norm SVM and the Fused SVM. The experiments on real-world datasets show that ISVMFD produces an interpretable classifier that is competitive with SVM in terms of classification ability and similar in computational times.

Algirdas, & Laukaitis (2008). Functional data analysis for cash flow and transactions intensity continuous-time prediction using hilbert-valued autoregressive processes. *European Journal of Operational Research*, *185*, 1607 – 1614.

Baesens, B., Mues, C., Martens, D., & Vanthienen, J. (2009). 50 years of data mining and or: upcoming trends and challenges. *Journal of the Operational Research Society*, *60*.

Baesens, B., Setiono, R., Mues, C., & Vanthienen, J. (2003). Using neural network rule extraction and decision tables for credit-risk evaluation. *Management Science*, *49*, 312–329.

Baíllo, A., Cuesta-Albertos, J., A., & Cuevas (2010). Supervised classification for a family of gaussian functional models. *Scandinavian Journal of Statistics*, *forthcoming, Arxiv preprint arXiv:1004.5031, 2010*.

Baíllo, A., & Grané, A. (2009). Local linear regression for functional predictor and scalar response. *Journal of Multivariate Analysis*, *100*, 102–111.

Barakat, N., & Diederich, J. (2006). Eclectic rule-extraction from support vector machines. *International Journal of Computational Intelligence*, *2*, 59–62.

Bradley, P., & Mangasarian, O. (1998). Feature selection via concave minimization and support vector machines. In *Proc. Fifteenth Int. Conf. Machine Learning* (pp. 82–90). San Francisco, CA: Morgan Kaufmann.

Candes, E., & Tao, T. (2007). The Dantzig selector: Statistical estimation when p is much larger than n. *The Annals of Statistics*, *35*, 2313–2351.

Cardot, H., & Sarda, P. (2005). Estimation in generalized linear models for functional data via penalized likelihood. *Journal of Multivariate Analysis*, *92*, 24–41.

Carrizosa, E., Martin-Barragan, B., & Morales, D. R. (2010). Binarized support vector machines. *INFORMS Journal on Computing*, *22*, 154–167.

Carrizosa, E., Martin-Barragan, B., & Morales, D. R. (2011). Detecting relevant variables and interactions in supervised classification. *European Journal of Operational Research*, *213*, 260–269.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*, 273–297.

Cuevas, A., Febrero, M., & Fraiman, R. (2002). Linear functional regression: The case of fixed design and functional response. *Canadian Journal of Statistics*, *30*, 285300.

Dauxois, J., Pousse, A., & Romain, Y. (1982). Asymptotic theory for the principal component analysis of a vector random function: some applications to statistical inference. *Journal of Multivariate Analysis*, *12*, 136–154.

Faraway, J. (1997). Regression analysis for a functional response. *Technometrics*, *39*, 254–261.

Ferraty, F., & Vieu, P. (2003). Curves discrimination: a nonparametric functional approach. *Computational Statistics and Data Analysis*, *44*, 161–173.

Ferraty, F., & Vieu, P. (2006). *Nonparametric functional data analysis: theory and practice*. Springer.

González, J., & Muñoz, A. (2010). *Representing Functional Data in Reproducing Kernel Hilbert Spaces with applications to clustering and classification*. Technical Report 013 Statistics and Econometrics Series. Ws102713.

Hastie, T., Tibshirani, R., & Friedman, J. (2001). *Elements of Statistical Learning: Data Mining, Inference and Prediction*. New York: Springer-Verlag.

Herbrich, R. (2002). *Learning Kernel Classifiers. Theory and Algorithms*. MIT Press.

James, G., Wang, J., & Zhu, J. (2009). Functional linear regression that's interpretable. *The Annals of Statistics*, *37*, 2083–2108.

James, G. M. (2002). Generalized linear models with functional predictors. *Journal of the Royal Statistical Society, Series B*, (pp. 411–432).

James, G. M., & Hastie, T. J. (2001). Functional linear discriminant analysis for irregularly sampled curves. *Journal of the Royal Statistical Society Series B*, *63*, 533–550.

Laukaitis, A., & Rackauskas, A. (2005). Functional data analysis for clients segmentation tasks. *European Journal of Operational Research*, *163*, 210 – 216. ¡ce:title¿Financial Modelling and Risk Management¡/ce:title¿.

Leng, X., & Müller, H. (2005). Classification using functional data analysis for temporal gene expression data. *Bioinformatics*, *22*, 68–76.

Lessmann, S., & Voß, S. (2009). A reference model for customer-centric data mining with support vector machines. *European Journal of Operational Research*, *199*, 520–530.

Li, B., & Yu, Q. (2008). Classification of functional data: A segmentation approach. *Comput. Stat. Data Anal.*, *52*, 4790–4800.

Li, J., Wei, L., Li, G., & Xu, W. (2011). An evolution strategy-based multiple kernels multi-criteria programming approach: The case of credit decision making. *Decision Support Systems*, *51*, 292–298.

Liang, K., & Zeger, S. (1986). Longitudinal data analysis using generalized linear models. *Biometrika*, *73*, 13–22.

Lindquist, A., & McKeague, I. (2009). Logistic regression with brownian-like predictors. *Journal of the American Statistical Association*, *104*, 1575–1585.

Lopez-Pintado, S., & Romo, J. (2009). On the concept of depth for functional data. *Journal of the American Statistical Association*, *104*, 718–734.

Martens, D., Baesens, B., & Van Gestel, T. (2009). Decompositional rule extraction from support vector machines by active learning. *IEEE Transactions on Knowledge and Data Engineering*, *21*, 178–191.

Martens, D., Baesens, B., Van Gestel, T., & Vanthienen, J. (2007). Comprehensible credit scoring models using rule extraction from support vector machines. *European Journal of Operational Research*, *183*, 1466–1476.

Moguerza, J., & Muñoz, A. (2006). Support vector machines with applications. *Statistical Science*, *21*, 322–336.

Muñoz, A., & González, J. (2010). Representing functional data using support vector machines. *Pattern Recognition Letters*, *31*, 511–516.

Pedroso, J., & Murata, N. (2001). Support vector machines with different norms: motivation, formulations and results. *Pattern Recognition Letters*, *22*, 1263–1272.

Ramsay, J., & Dalzell, C. (1991). Some tools for functional data analysis (with discusion). *J. R. Stat. Soc. Ser. B*, *53*, 539–572.

Ramsay, J., & Silverman, B. (2002). *Applied Functional Data Analysis*. New York: Springer-Verlag.

Ramsay, J., & Silverman, B. (2005). *Functional Data Analysis*. New York: Springer-Verlag.

Rapaport, F., Barillot, E., & Vert, J. (2008). Classification of arrayCGH data using fused SVM. *Bioinformatics*, *24*, 375–382.

Rossi, F., & Villa, N. (2006). Support vector machines for functional data analysis. *Neurocomputing*, *69*, 730–742.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *J. R. Stat. Soc. Ser. B*, *58*, 267–288.

Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., & Knight, K. (2005). Sparsity and smoothness via the fused lasso. *Journal Of The Royal Statistical Society Series B*, *67*, 91–108.

Van Gestel, T., Martens, D., Baesens, B., Feremans, D., Huysmans, J., & Vanthienen, J. (2007). Forecasting and analyzing insurance companies' ratings. *International Journal of Forecasting*, *23*, 513–529.

Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. Springer-Verlag.

Verbeke, W., Dejaeger, K., Martens, D., Hur, J., & Baesens, B. (20011). New insights into churn prediction in the telecommunication sector: A profit driven data mining approach. *European Journal of Operational Research*, *In press*.

Zhu, J., Rosset, S., Hastie, T., & Tibshirani, R. (2003). 1-norm support vector machines. In *Neural Information Processing Systems* (pp. 49–56). volume 16.

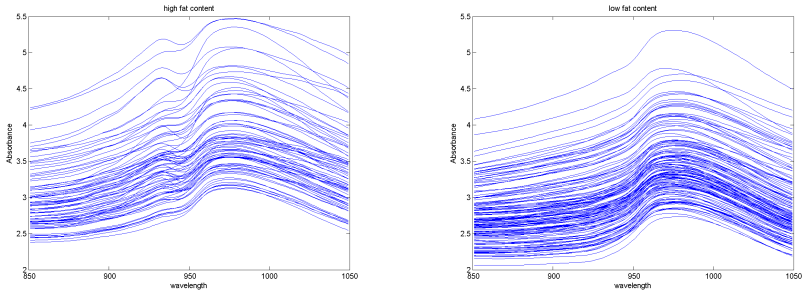

Figure 1: Representation of original data for `tecator` dataset.

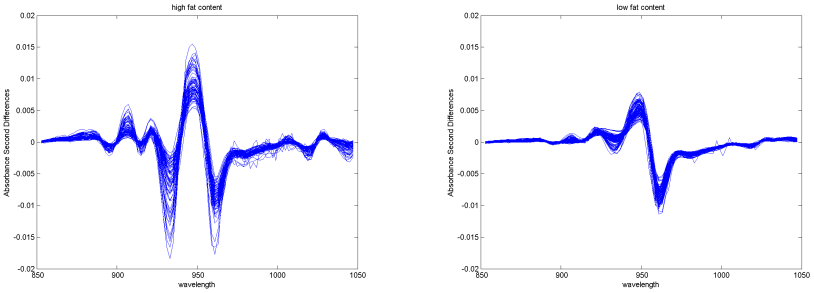

Figure 2: Representation of derivatives of the curves for `tecator` dataset.

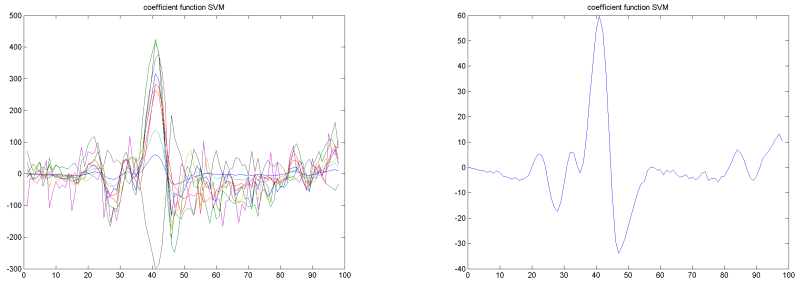

Figure 3: Coefficient functions of SVM for `tecator` dataset.

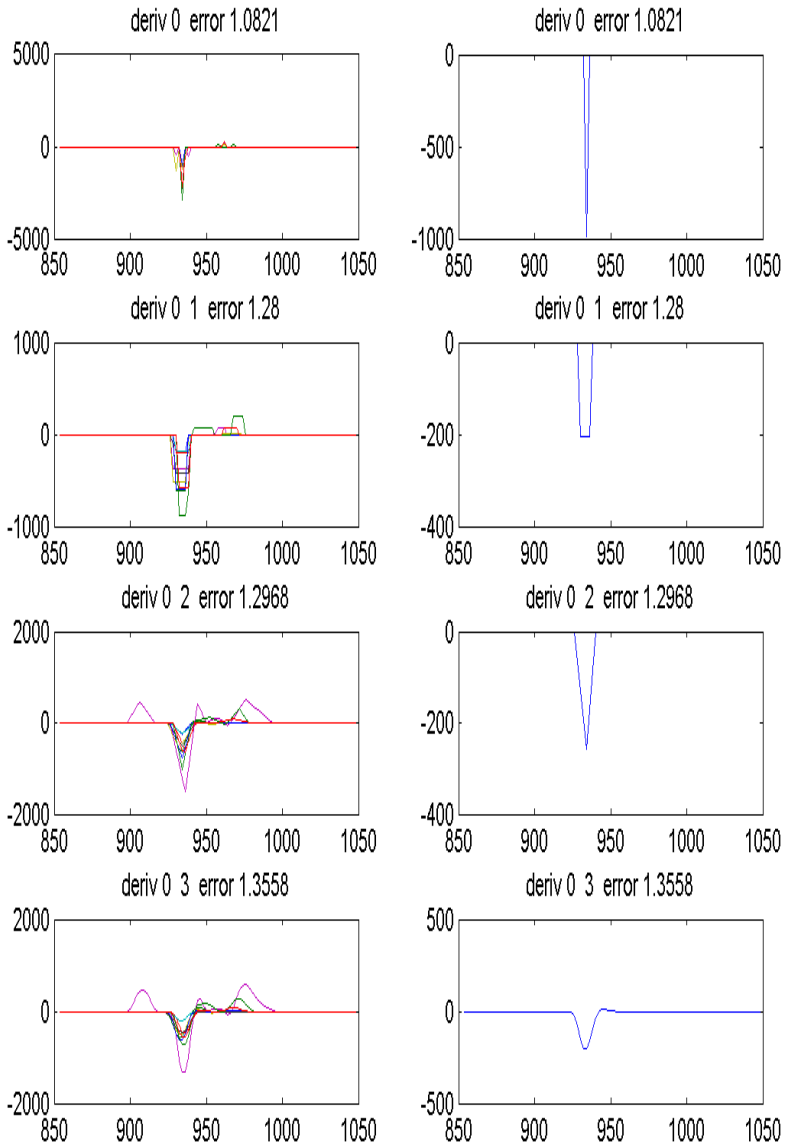

Figure 4: Coefficient functions of ISVMFD for `tecator` dataset. Part I.

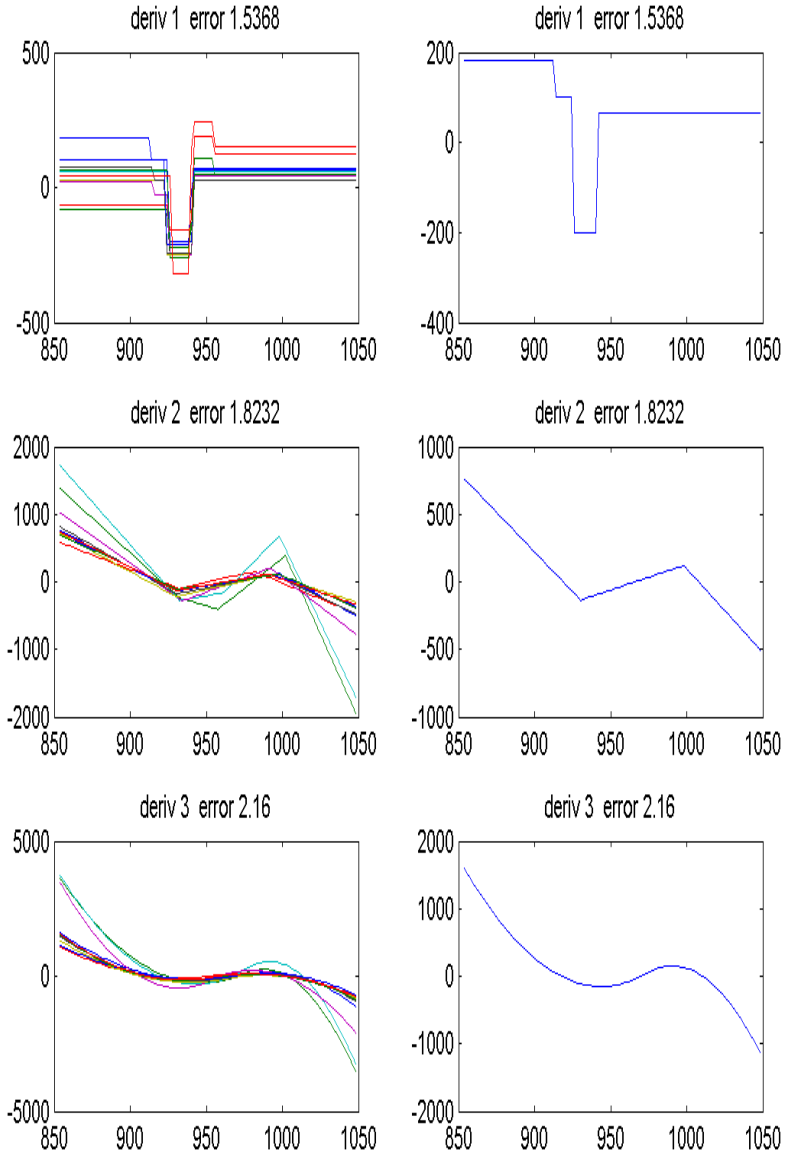

Figure 5: Coefficient functions of ISVMFD for `tecator` dataset. Part II.

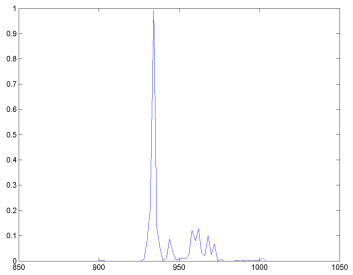

Figure 6: Proportion of times that every channel is detected as important by the classifier `tecator` dataset.

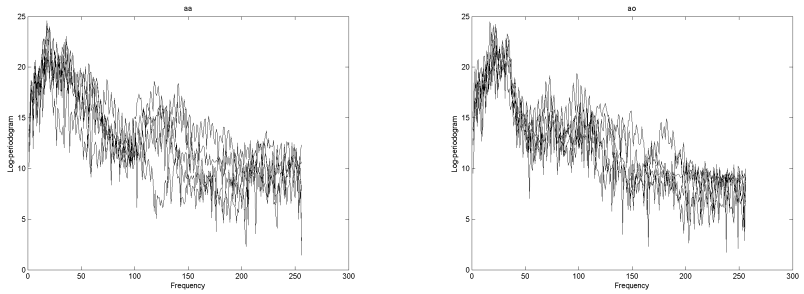

Figure 7: Representation of data for `phoneme` dataset.

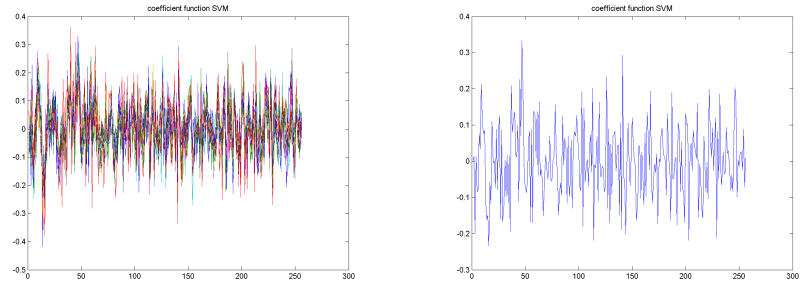

Figure 8: Coefficient functions of SVM for `phoneme` dataset.

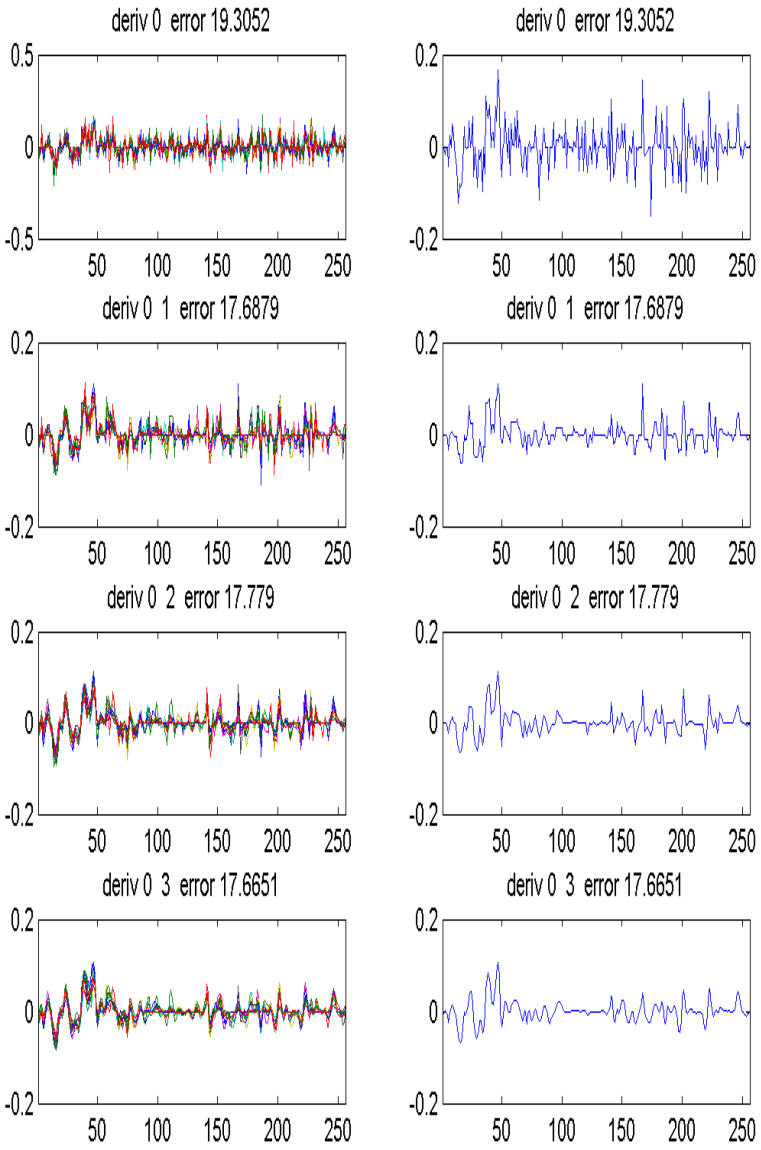

Figure 9: Coefficient functions of ISVMFD for `phoneme` dataset. Part I.

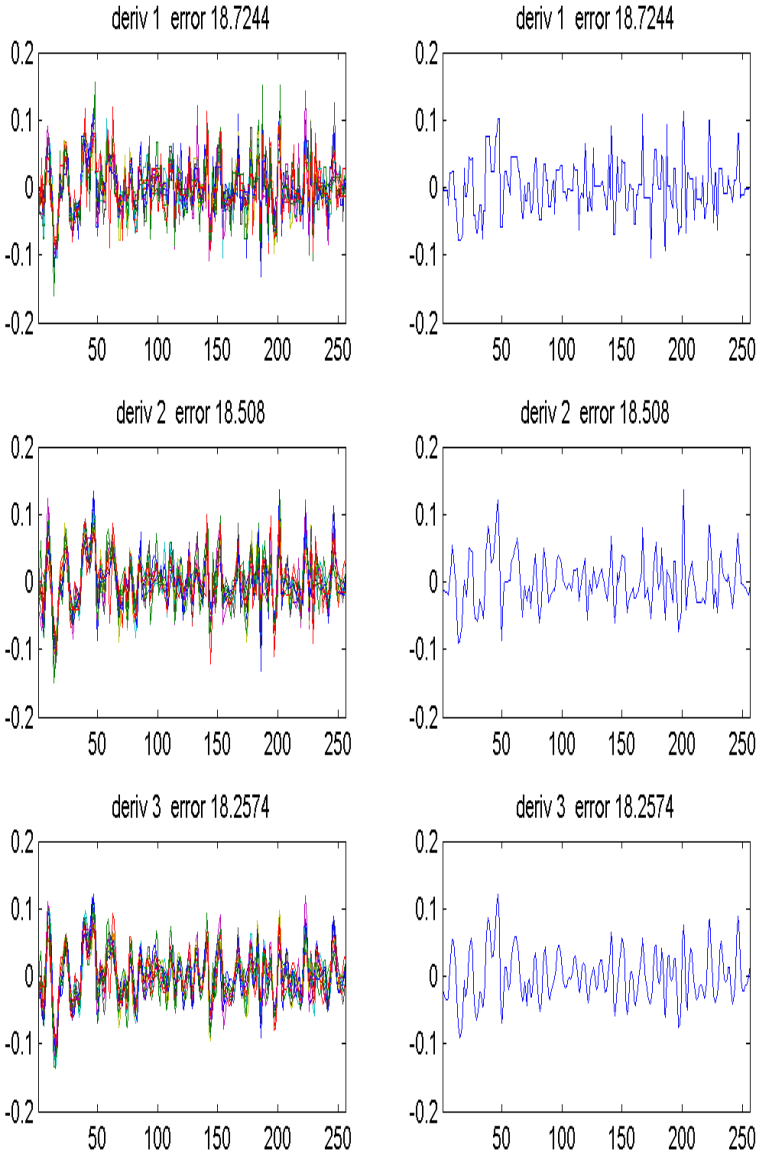

Figure 10: Coefficient functions of ISVMFD for `phoneme` dataset. Part II.

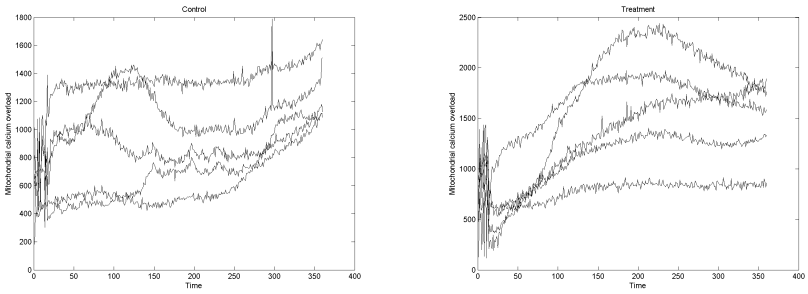

Figure 11: Representation of data for `ca` dataset.

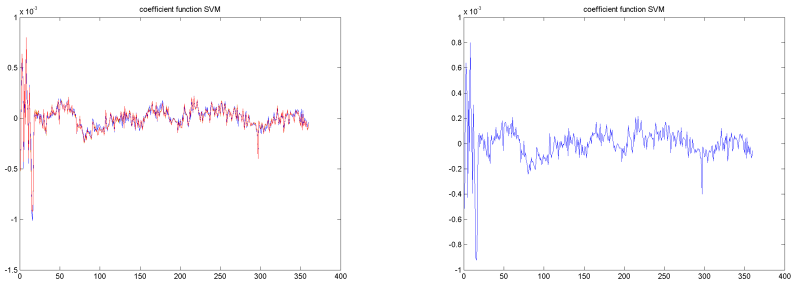

Figure 12: Coefficient functions of SVM for `ca` dataset.

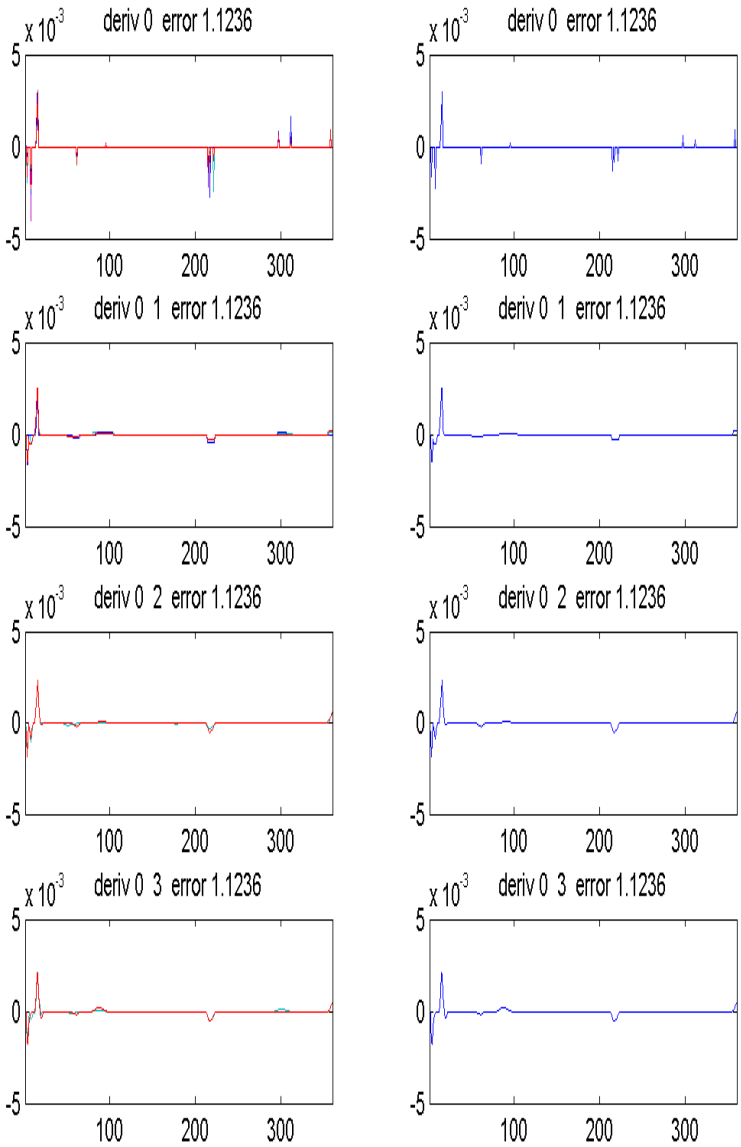

Figure 13: Coefficient functions of ISVMFD for `ca` dataset. Part I.

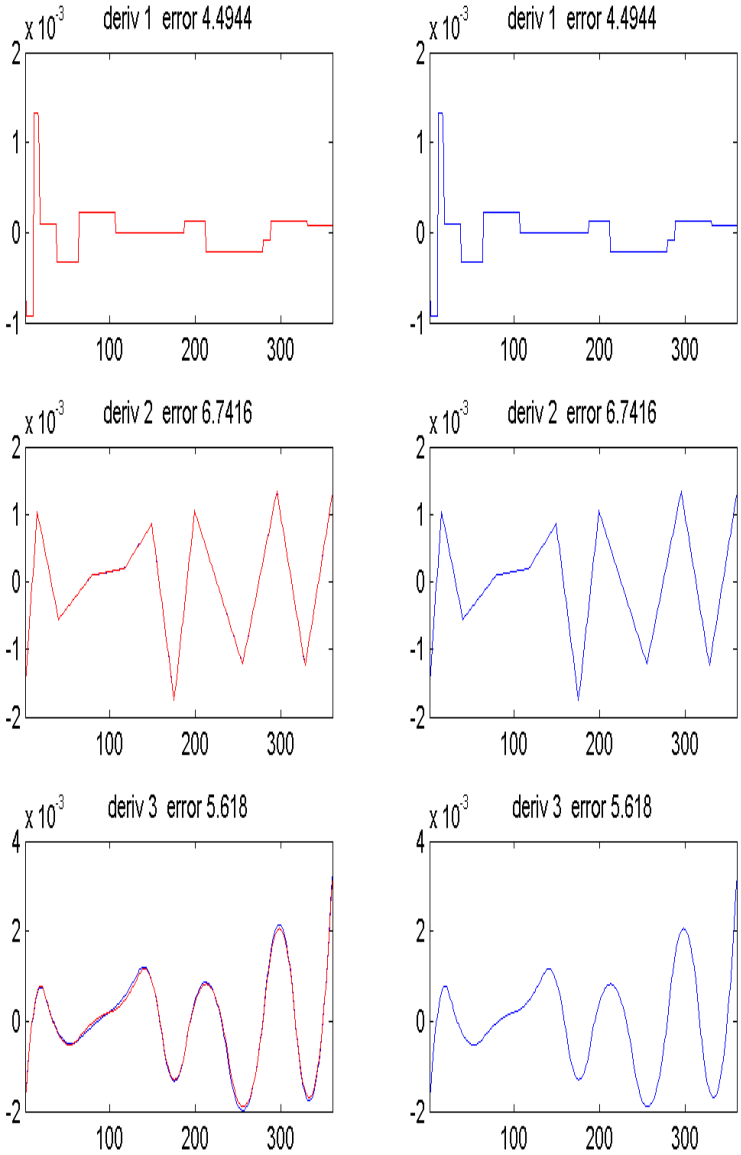

Figure 14: Coefficient functions of ISVMFD for `ca` dataset. Part II.

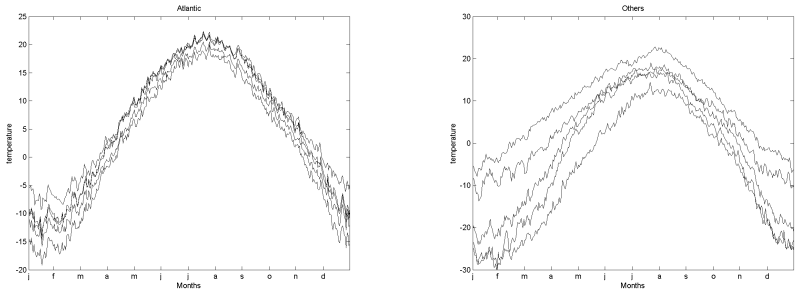

Figure 15: Representation of data for `Regions` dataset.

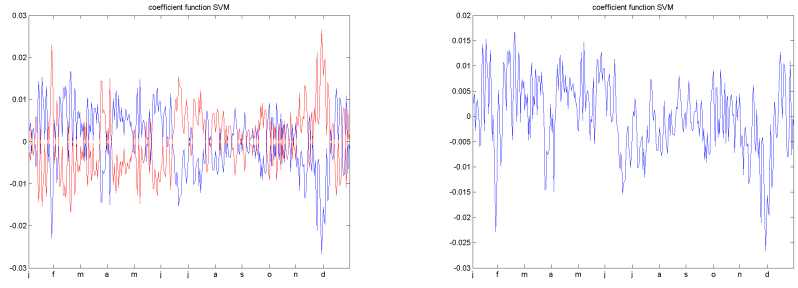

Figure 16: Coefficient functions of SVM for `Regions` dataset.

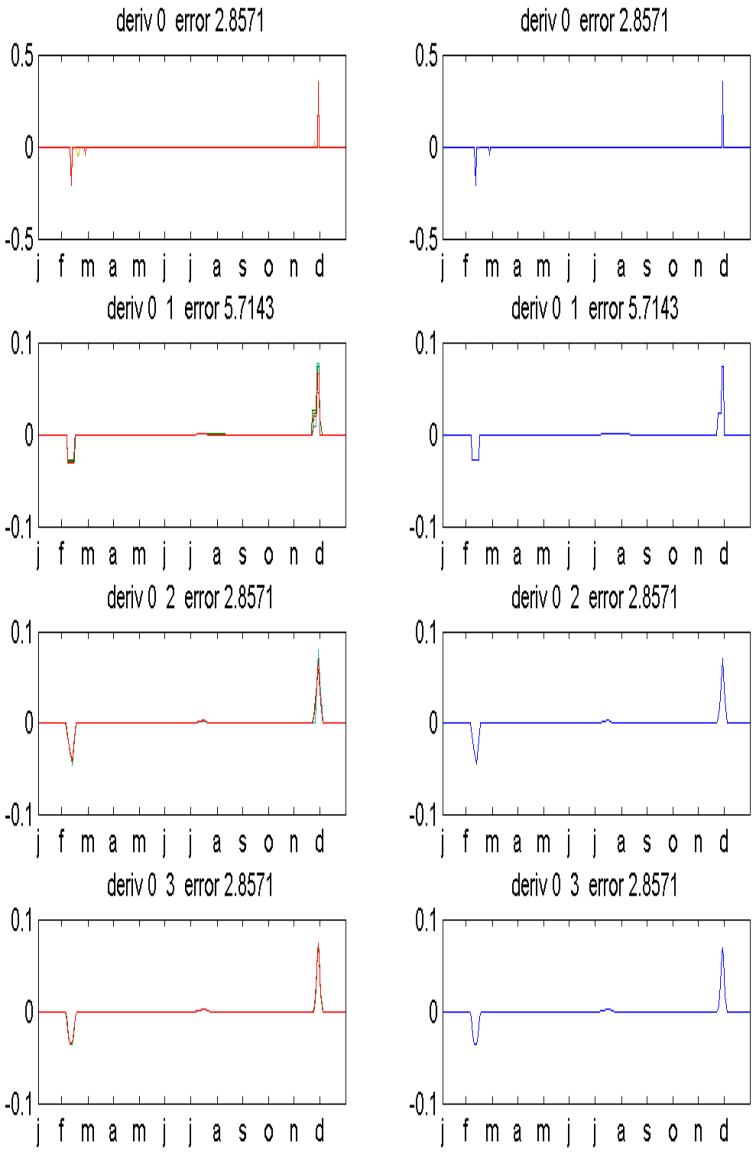

Figure 17: Coefficient functions of ISVMFD for `Regions` dataset. Part I.

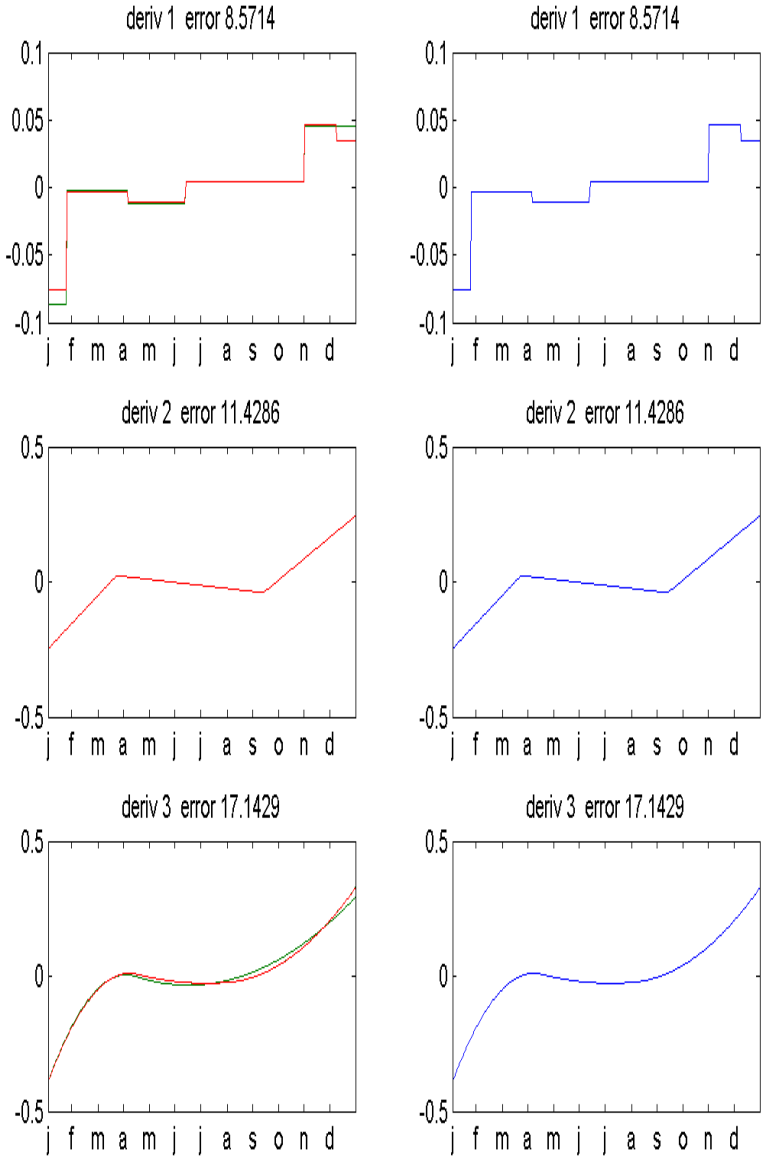

Figure 18: Coefficient functions of ISVMFD for `Regions` dataset. Part II.

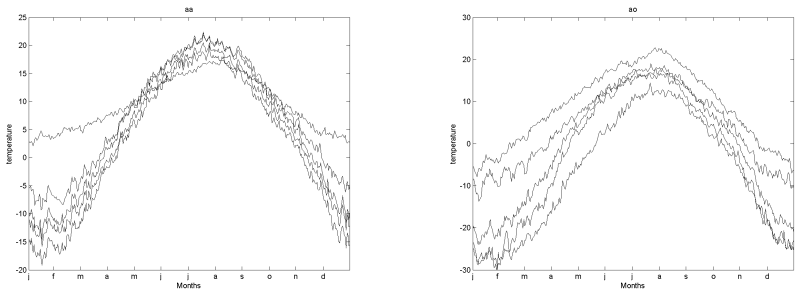

Figure 19: Representation of data for `Rain` dataset.

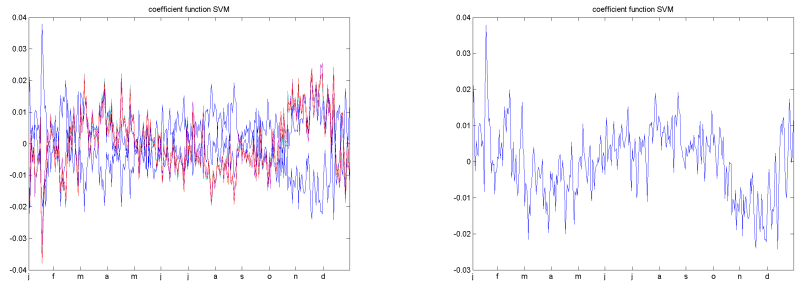

Figure 20: Coefficient functions of SVM for `Rain` dataset.

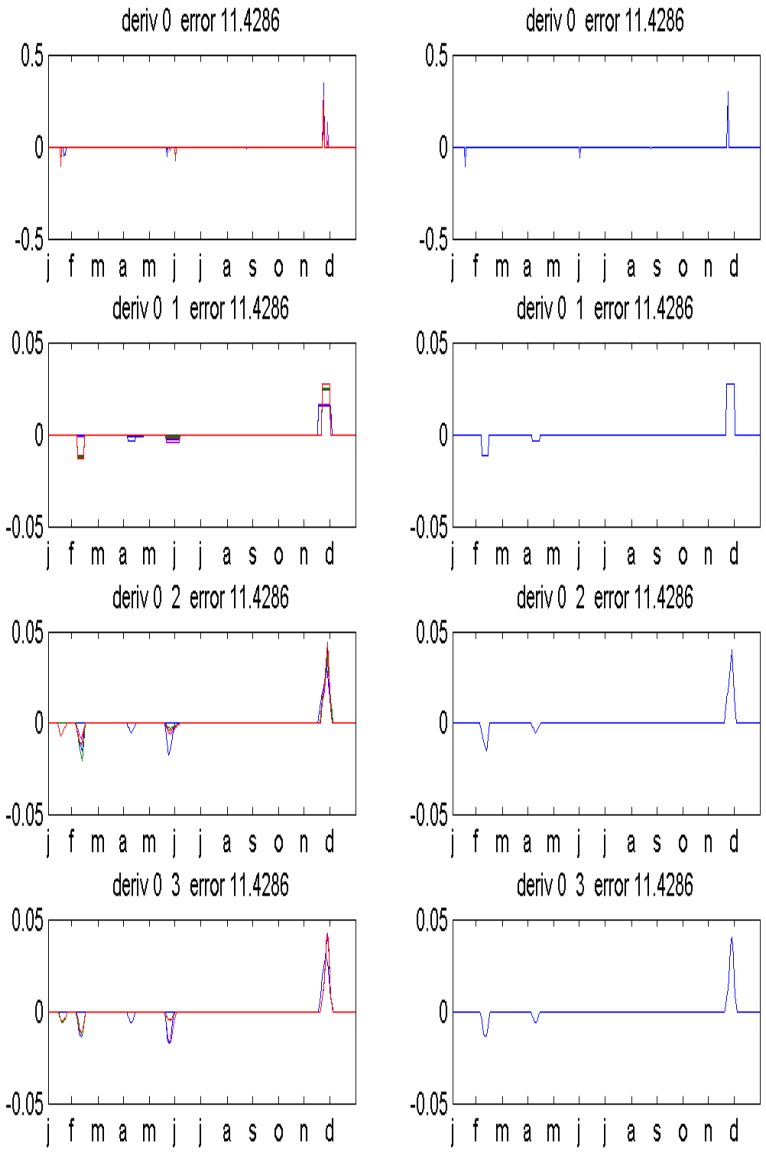

Figure 21: Coefficient functions of ISVMFD for `Rain` dataset. Part I.

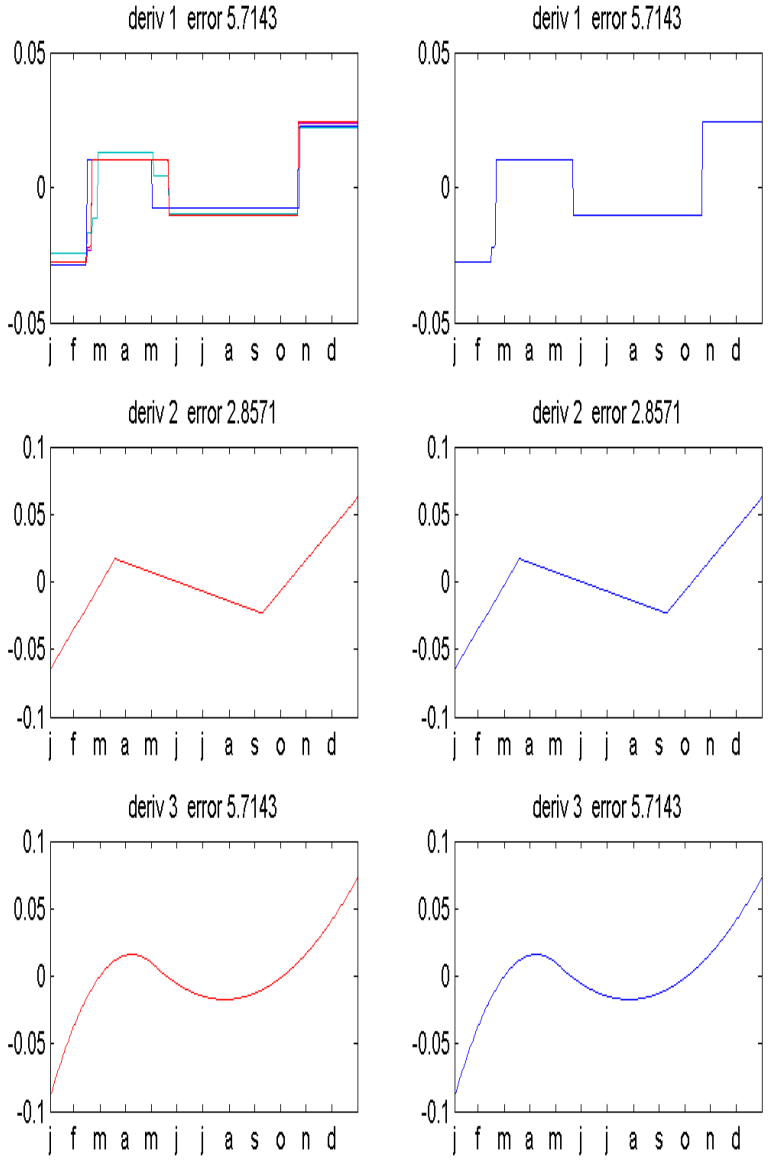

Figure 22: Coefficient functions of ISVMFD for `Rain` dataset. Part II.