

Variable Projection Support Vector Machines and Some Applications Using Adaptive Hermite Expansions

Tamás Dózsa 

*Department of Numerical Analysis
HUN-REN Institute for Computer Science and Control
Eötvös Loránd University, Budapest H-1111, Hungary
dozsatamas@sztki.hu; dotuaai@inf.elte.hu;
www.sztki.hu/dozsa-tamas-gabor*

Federico Deuschle and Bram Cornelis
*Siemens Digital Industries Software, 68 Interleuvenlaan
KU Leuven, Department of Mechanical Engineering
Leuven B-3001, Belgium*

Péter Kovács *

*Department of Numerical Analysis, Eötvös
Loránd University, Pázmány Péter sétány 1/C
Budapest 1117, Hungary
kovika@inf.elte.hu*

Received 19 April 2023

Accepted 25 October 2023

Published Online 11 December 2023

In this paper, we develop the so-called variable projection support vector machine (VP-SVM) algorithm that is a generalization of the classical SVM. In fact, the VP block serves as an automatic feature extractor to the SVM, which are trained simultaneously. We consider the primal form of the arising optimization task and investigate the use of nonlinear kernels. We show that by choosing the so-called adaptive Hermite function system as the basis of the orthogonal projections in our classification scheme, several real-world signal processing problems can be successfully solved. In particular, we test the effectiveness of our method in two case studies corresponding to anomaly detection. First, we consider the detection of abnormal peaks in accelerometer data caused by sensor malfunction. Then, we show that the proposed classification algorithm can be used to detect abnormalities in ECG data. Our experiments show that the proposed method produces comparable results to the state-of-the-art while retaining desired properties of SVM classification such as light weight architecture and interpretability. We implement the proposed method on a microcontroller and demonstrate its ability to be used for real-time applications. To further minimize computational cost, discrete orthogonal adaptive Hermite functions are introduced for the first time.

Keywords: Support vector machines; Hermite functions; variable projection; ECG classification; anomaly detection.

*Corresponding author.

This is an Open Access article published by World Scientific Publishing Company. It is distributed under the terms of the Creative Commons Attribution 4.0 (CC BY) License which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

1. Introduction

Machine Learning (ML) algorithms are often used to solve regression or classification problems. They usually take a large amount of input data and “learn” a mapping through an optimization process known as training, which pairs the input data samples to predefined targets. Finding an appropriate representation of the input data is an important part of solving any problem in ML and can greatly influence the generalization potential of the optimized ML model.^{1,2} The data samples acting as the input to the ML algorithm are obtained by applying a series of transformations known as feature extraction steps to the original data.

Traditionally, these transformations were chosen in an intuitive manner based on *a priori* knowledge about the problem. Popular examples of feature extraction done in this way include principal component analysis (PCA),³ the discrete Fourier Transform (DFT)⁴ and various time frequency representations of the data obtained for example through wavelet or Gabor transformations.⁴ More recently, semiadaptive transformations, where the parameters of the feature extraction scheme were chosen to ensure a good approximation of the original data were investigated in Refs. 5–7. Despite their advantages, the performance of these approaches is still suboptimal, since the optimization of the feature extraction pipeline and the training of the underlying ML model are separated.

The introduction of convolutional neural networks (CNNs) led to the rise of ML algorithms, where appropriate input representation was learned *alongside* the weights of the classification or regression model.^{1,2,8} CNNs implement discrete convolutions as layers of a neural network, where the values (weights) of the convolution kernel are treated as free parameters. This idea led to the development of many new applications especially in image processing.^{2,8} One of the most well-known limitations of convolution layers is the fact that the optimized parameters are difficult to interpret by humans.⁹ This can be problematic for some applications (for example, in medical signal processing or autonomous driving), where an explanation is required for the output of the ML model.

Recently, in addition to CNNs, mathematically justified ML methods have been introduced capable

of automatic feature extraction. For example, Neural Dynamic Classification¹⁰ (NDC) proposes a transformation of the input vectors that guarantees an increase in the probability of correct classification.

In order to understand black-box ML models and to explain how variables are being combined to make predictions, many analysis tools were proposed.⁹ Besides that, ML approaches consisting of interpretable parameters were intensively researched in pursuit of fully explainable artificial intelligence (XAI) systems.¹¹ Recent advances in this area include the introduction of deep unfolding networks¹² as well as techniques like local interpretable model agnostic explanations¹³ which can be used to provide explanations for ML model predictions.

Recently, physics informed machine learning methods have also enjoyed growing popularity. These methods aim to incorporate traditional physical models into ML algorithms to achieve a degree of explainability. Methods range from domain-specific models¹⁴ to approaches that are designed for general classification and regression tasks. These include ML approaches building on mathematical knowledge about differential equations such as ODENet¹⁵ and FEMa.¹⁶ FEMa, for example, proposes an interesting classification algorithm rooted in the finite element method.

Another recently developed approach introduced variable projection networks (VP-NET),¹⁷ where the first few layers of a fully connected neural network implemented adaptive orthogonal transformations with free parameters to learn appropriate data representation. If the transformations implemented in these so-called VP-layers are chosen carefully, then the learned parameters will be interpretable to humans.¹⁷

Recently, classification problems in medical signal processing and autonomous vehicle control were successfully addressed using approaches based on VP-NET.^{18,19} These problems often require real-time evaluation of the trained model on hardware with limited resources such as microcontrollers or field programmable gate arrays (FPGAs). Optimal data representations provided by variable projection transformations allow for ML solutions that do not involve deep neural networks. In this work, we propose an extension of the classical Support Vector Machine (SVM) called variable projection support

vector machine (VP-SVM) in Ref. 20. In particular, we extended the primal form of the objective functions of SVMs with adaptive orthogonal projections and showed how stochastic subgradient descent (SSGD) can be used to train the proposed classifier.

In this paper, we further extend our results in Ref. 20. We provide deeper explanations for the properties of the proposed VP-SVM classification scheme and supplement it with new theoretical and experimental results. In our current investigation, we focus on solving classification tasks, where the input samples are considered to be 1D signals and the data transformation part of VP-SVM is realized by the orthogonal expansion of the input data using adaptive Hermite functions.²¹ Since their introduction in Ref. 21, adaptive Hermite functions have played a crucial role in many signal processing applications.^{6,18,19,21} In this work, we show how the adaptive Hermite function-based VP-SVM classifier can be used to effectively solve two different, real-world classification tasks.

In our first application, we use VP-SVM to detect peaks from accelerometer data which were caused by sensor malfunction. We show that VP-SVM can be used to detect such peaks, furthermore showcase how the output and learned parameters of the underlying VP transformations can be interpreted.

Finally, we investigate the effectiveness of the proposed VP-SVM classifier through a biomedical signal processing task. Namely, we solve the problem of recognizing ventricular ectopic beats (VEBs) in electrocardiogram (ECG) signals. VEBs are one of the abnormal heartbeat classes recommended in the ANSI/AAMI EC57:1998 standard and can be found in the MIT-BiH publicly available database abundantly.²² This allows for the partial comparison of the proposed method to the state-of-the-art.²³

In addition to the investigated applications, we propose to use an orthogonal VP transformation based on the discrete orthogonal variation of the adaptive Hermite function system. We point out, how the use of discrete orthogonal bases with VP-SVM can greatly reduce the number of computations needed for data transformation, further decreasing the reliance of VP-SVM on computational resources.

The rest of this paper is structured as follows. In Sec. 2, we review adaptive orthogonal transformations, as well as classical support vector machines.

We also provide an overview of the proposed VP-SVM classifier. In Sec. 3, we discuss adaptive orthogonal Hermite expansions and propose discrete orthogonal adaptive Hermite functions. In Sec. 4, we introduce the above-mentioned real-world applications, discuss experimental setups and comparisons with the state of the art. Finally, Sec. 5 contains our conclusions and future plans.

2. Variable Projection Support Vector Machines

2.1. Variable projection operators

Let $\mathbf{f} \in \mathbb{R}^N (N \in \mathbb{N})$ be an N point sampling of an $f \in L_2(\mathbb{R})$ signal (a single data example). In the proposed method, we are going to extract $n \in \mathbb{N}$ number of features from $\mathbf{f}$ by

$$\mathbf{f} \approx \tilde{\mathbf{f}} = P_{\Phi(\boldsymbol{\eta})} \mathbf{f} := \Phi(\boldsymbol{\eta})(\Phi(\boldsymbol{\eta})^+ \mathbf{f}), \quad (1)$$

where $\Phi(\boldsymbol{\eta}) \in \mathbb{R}^{N \times n} (N \gg n)$. In (1), $\Phi(\boldsymbol{\eta})^+$ refers to the Moore–Penrose pseudo inverse of $\Phi(\boldsymbol{\eta})$ and $\boldsymbol{\eta} \in \mathbb{R}^M (M \in \mathbb{N})$ is a parameter vector that specifies $\Phi(\boldsymbol{\eta})$. Thus, the approximation $\tilde{\mathbf{f}}$ is a projection onto the column space of $\Phi(\boldsymbol{\eta})$. This space is usually spanned by the discrete sampling of an $L_2(\mathbb{R})$ basis. The functions $\varphi_0^\eta, \dots, \varphi_{n-1}^\eta \in L_2(\mathbb{R})$, whose samplings make up the columns of $\Phi(\boldsymbol{\eta})$, depend on the parameters in $\boldsymbol{\eta}$ in a nonlinear fashion. Furthermore, we will assume that the partial derivatives of $\varphi_k^\eta (k = 0, \dots, n-1)$ exist with respect to $\boldsymbol{\eta}$. As discussed in Sec. 3, the actual choice of these functions in this paper satisfies the above conditions. Nonetheless, it is important to note that other function systems can also be used to define the matrix $\Phi(\boldsymbol{\eta})$.

The projection operator $P_{\Phi(\boldsymbol{\eta})}$ in (1) is referred to as a variable projection operator.^{7,24} Once the parametrized family of functions $\varphi_k^\eta (k = 0, \dots, n-1)$ has been chosen for a given signal $\mathbf{f}$, we can determine the best parameters $\boldsymbol{\eta}$ by solving

$$\min_{\boldsymbol{\eta} \in \mathbb{R}^M} r_2(\boldsymbol{\eta}, \mathbf{f}) = \min_{\boldsymbol{\eta} \in \mathbb{R}^M} \|\mathbf{f} - P_{\Phi(\boldsymbol{\eta})} \mathbf{f}\|_2^2. \quad (2)$$

Golub and Pereyra were the first to show in Ref. 24 that the gradient of the functional r_2 can be analytically calculated provided that the partial derivatives of $\Phi(\boldsymbol{\eta})$ are known. This allows us to determine the optimal parameters $\boldsymbol{\eta}$ using gradient descent-based algorithms.

Solving (2) will be an important part of specifying the data transformation (feature extraction) step implemented in our variation of the SVM classifier. Once the optimal $\boldsymbol{\eta}$ parameters have been determined, (depending on the task at hand) we can represent the input signal $\mathbf{f}$ with one of the following transformations:

- The transformation

$$T_0^\eta(\mathbf{f}) := \Phi(\boldsymbol{\eta})^+ \mathbf{f} =: \mathbf{c} \in \mathbb{R}^n \quad (3)$$

can be interpreted as a feature extraction and dimension reduction step.

- In the presented applications (see Sec. 4), the columns of $\Phi(\boldsymbol{\eta})$ consist of discrete samplings of smooth functions $\varphi_0^\eta, \dots, \varphi_{n-1}^\eta \in L_2(\mathbb{R})$. Furthermore, as k increases φ_k^η will display rapidly oscillatory behavior. Hence, the smooth approximation

$$T_1^\eta(\mathbf{f}) := P_{\Phi(\boldsymbol{\eta})} \mathbf{f} = \Phi(\boldsymbol{\eta}) \Phi(\boldsymbol{\eta})^+ \mathbf{f} \quad (4)$$

can be interpreted as the result of low pass filtering using the basis functions φ_k^η .

- Taking the orthogonal complement

$$T_2^\eta(\mathbf{f}) := P_{\Phi(\boldsymbol{\eta})}^\perp \mathbf{f} = \mathbf{f} - P_{\Phi(\boldsymbol{\eta})} \mathbf{f} \quad (5)$$

of the variable projection operator (and considering again the properties of the basis functions used in our work) removes low frequency components of $\mathbf{f}$ and thus can be interpreted as a high pass filter.

The choice of the appropriate transformation given the matrix $\Phi(\boldsymbol{\eta})$ depends on the task to be solved. Extending our previous work,²⁰ we propose real-world applications in Sec. 4 that depend on variable projection support vector machines using transformation (5).

2.2. VP-SVM objectives

In this section, we give a brief review of SVM and VP-SVM classification methods. More precisely, we are going to examine the classical support vector machine classifier and extend it with variable projections.

Let $\mathcal{S} := \{(\mathbf{f}_i, y_i)\} \subset \mathcal{X} \times \{-1, 1\}$ ($i = 1, \dots, q$) be a labeled set of data with a number of $q \in \mathbb{N}$ examples. For this work, we assume that $\mathcal{X} \subset \mathbb{R}^N$ ($N \in \mathbb{N}$). The support vector machine (SVM) method aims to identify a mapping $\mathcal{F} : \mathcal{X} \rightarrow \{-1, 1\}$, for which $\mathcal{F}(\mathbf{f}_i) = y_i$, $(\mathbf{f}_i, y_i) \in \mathcal{S}$. SVM does this,

by finding an optimal hyperplane in $\mathcal{X}$, which separates the data examples $\mathbf{f}_i$ by class labels y_i .

We note that SVM can be extended for multiclass classification problems. This is usually done by training several binary SVM classifiers, then combining their outputs to solve the multiclass problem. Common strategies include the one-versus-all approach with winner-takes-all output combination,²⁵ one-versus-one methods using a voting scheme²⁵ and a technique known as pairwise coupling.²⁵ We note that even though in theory any of the above-mentioned techniques would work with the proposed VP-SVM classifiers, finding the best technique for VP-SVM multiclass problems is outside the scope of this work and will only be considered in our future research. The codebase of this research is also published as an opensource gitLab repository,²⁶ which can be utilized for such future research tasks.

Usually, finding the optimal hyperplane is posed as the linear programming problem

$$\begin{aligned} \min_{\mathbf{w} \in \mathbb{R}^N, b \in \mathbb{R}, \xi \in \mathbb{R}^q} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{k=1}^q \xi_k \\ \text{subject to} \quad & y_k(\mathbf{w}^T \mathbf{f}_k + b) \geq 1 - \xi_k, \quad (k = 1, \dots, q). \end{aligned} \quad (6)$$

Equation (6) is referred to as the primal form of the soft margin support vector classification problem and solved using convex optimization tools (see e.g. Refs. 27–29). In practice, the dual formulation of (6) is often used instead, as it provides a convenient way to deal with the above constraints and allows for the expression of the problem in terms of the inner products of the data examples $\mathbf{f}_k$.

Despite the advantages of the dual formulation, in this work we opted to use the unconstrained, equivalent formulation of the primal SVM problem (6). Solving this problem is equivalent to minimizing

$$\begin{aligned} C \cdot \sum_{i=1}^q \max(0, 1 - y_i \cdot (\mathbf{w}^T \mathbf{f}_i + b)) + \|\mathbf{w}\|_2^2 \\ (\mathbf{w} \in \mathbb{R}^N, C, b \in \mathbb{R}) \end{aligned} \quad (7)$$

with respect to the variables $\mathbf{w}$ and b . Our reason for choosing the above formulation is the fact that it can be solved using gradient-based methods.^{27,30,31} This is preferable, as we already stated in Sec. 2.1 that the gradient of the orthogonal projection operators can also be calculated analytically.

We can easily extend (7) with an adaptive feature extraction step by

$$C \sum_{k=1}^q \max(0, 1 - y_k(\mathbf{w}^T T_i^\eta(\mathbf{f}_k) + b) + \|\mathbf{w}\|_2^2 + R(\boldsymbol{\eta}) \quad (i \in \{0, 1, 2\}). \quad (8)$$

We will refer to (8) as the objective function of the linear variable projection support vector machine (VP-SVM). In (8), T_i^η is one of the transformations defined in (3)–(5). Furthermore, in (8) we extended the original SVM objective (7) by the regulatory term $R(\boldsymbol{\eta})$ defined as

$$R(\boldsymbol{\eta}) := \frac{\alpha}{q} \sum_{k=1}^q \frac{\|\mathbf{f}_k - T_1^\eta(\mathbf{f}_k)\|_2^2}{\|\mathbf{f}_k\|_2^2} \quad (\alpha \in \mathbb{R}). \quad (9)$$

This regulatory term is important to ensure that the adaptive orthogonal projections approximate the data examples well. In addition, when the objective function (8) is minimized with respect to the free parameters $\mathbf{w}$, $\boldsymbol{\eta}$ and b using gradient-based methods, the term (9) helps us to mitigate the problem of vanishing gradients.

The popularity of SVM classifiers is in large part due to their ability to be used with so-called Mercer kernels.³⁰ The application of these kernels is equivalent to the transformation of the data examples to a high dimensional reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ and looking for a separating hyperplane in $\mathcal{H}$. The main advantage of this approach is that the transformed data examples in $\mathcal{H}$ often become more easily separable, thus allowing SVM to classify nonlinearly separable data. When posing the SVM optimization problem using Mercer kernels, usually the Wolfe-dual of (8) is considered

$$\begin{aligned} \max_{\mathbf{a} \in \mathbb{R}^q} \quad & \sum_{k=1}^q a_k - \frac{1}{2} \sum_{k=1}^q \sum_{j=1}^q a_k a_j y_k y_j \langle \mathbf{f}_k, \mathbf{f}_j \rangle \\ \text{subject to} \quad & \sum_{k=1}^q a_k y_k = 0, \quad L \geq a_k \geq 0 \\ & (L \in \mathbb{R}, k = 1, \dots, q), \end{aligned} \quad (10)$$

where the inner products $\langle \mathbf{f}_k, \mathbf{f}_j \rangle$ may be replaced with an appropriate kernel function.³⁰ Thus, the kernel function can be interpreted as the inner product of the space $\mathcal{H}$. As further discussed in Sec. 4, in our experiments we used the so-called radial basis function (RBF) kernel $k(\mathbf{f}, \mathbf{g}) := e^{-\|\mathbf{f} - \mathbf{g}\|_2^2 / \sigma^2}$ ($\mathbf{f}, \mathbf{g} \in \mathbb{R}^N, \sigma > 0$), however the proposed

nonlinear VP-SVM objectives may be used with any applicable Mercer kernel.

Despite the clear benefits of the dual formulation, the primal form of nonlinear SVM objective functions has been a well researched area^{27,30,31} with several important results. In Ref. 30 for example, the author argues that primal optimization is preferable for large-scale training data, because as q increases one is forced to use approximations of the objective by disregarding some examples when calculating the inner products in (10). There is no guarantee, however, that an approximate dual solution produces an acceptable approximate solution in the primal.

Furthermore, in the case of kernelized VP-SVM, the orthogonal transformations (3)–(5) are always applied directly to the data examples $\mathbf{f}$. This means that the kernel-defined inner products will take the form

$$k(T_i^\eta(\mathbf{f}_k), T_i^\eta(\mathbf{f}_j)) \quad (j, k = 1, \dots, q, i \in \{0, 1, 2\}),$$

thus any parameter update of $\boldsymbol{\eta}$ would require the re-evaluation of every inner product. In order to make the proposed method computationally feasible, instead of (10), we considered the primal form of the nonlinear SVM objective as proposed in Ref. 30

$$\min_{\mathbf{f} \in \mathcal{H}} \mu \|\mathbf{h}\|_{\mathcal{H}}^2 + \sum_{k=1}^q \max(0, 1 - y_k h(\mathbf{f}_k)), \quad (11)$$

where $\mu = 1/C$ and h denotes a mapping from $\mathbb{R}^N$ to $\mathcal{H}$.

Even though h is not known explicitly, the representer theorem (see e.g. Ref. 30) states that it can be expressed using the data examples $\mathbf{f}_j$ and the kernel function $k(\cdot, \cdot)$:

$$h(\mathbf{f}) = \sum_{j=1}^q \beta_j k(\mathbf{f}, \mathbf{f}_j) \quad (\beta \in \mathbb{R}^q). \quad (12)$$

By (12), it is possible to formulate the primal nonlinear SVM objective (11) via the kernel functions:

$$\begin{aligned} \mu \sum_{k,j=1}^q \beta_k \beta_j k(\mathbf{f}_k, \mathbf{f}_j) \\ + \sum_{k=1}^q \max \left(0, 1 - y_k \left(\sum_{j=1}^q k(\mathbf{f}_k, \mathbf{f}_j) \beta_j \right) \right). \end{aligned} \quad (13)$$

This objective function can be easily extended with the variable projection transformations discussed in

Sec. 2.1:

$$R(\boldsymbol{\eta}) + \mu \sum_{k,j=1}^q \beta_k \beta_j k(T_i^\eta(\mathbf{f}_k), T_i^\eta(\mathbf{f}_j)) + \sum_{k=1}^q \max \left(0, 1 - y_k \left(\sum_{j=1}^q k(T_i^\eta(\mathbf{f}_k), T_i^\eta(\mathbf{f}_j)) \beta_j + b \right) \right), \quad (14)$$

where $R(\boldsymbol{\eta})$ is the regulatory term defined in (9). It is also important to note that the parameters $\boldsymbol{\beta} \in \mathbb{R}^q$ in (14) are different from the dual parameters $\mathbf{a} \in \mathbb{R}^q$ in (10).³⁰ At first glance, it may not seem obvious from (14) why the primal formulation is more efficient from a computational point of view. If however, we consider Chapelle's observation³⁰ that approximate dual solutions are not guaranteed to yield a good primal approximate solution together with our stated desire to use stochastic gradient descent-based optimization to minimize the SVM objective, we can derive approximate objective functions from (14) with much reduced computational costs as detailed in the following section.

2.3. Minimizing the VP-SVM objectives

In this section, we discuss how to minimize the VP-SVM objectives (8) and (14) using stochastic gradient descent (SGD) and provide the gradients with respect to the trainable parameters. The first obstacle when applying SGD to minimize the above-mentioned objectives is that neither (8) nor (14) is differentiable everywhere, as clearly the gradients with respect to the primal weight and bias parameters do not exist at 0.

This problem can be overcome by replacing the gradients with their *subgradients* in the points, where the gradient does not exist. For example, if $1 - y_k(\mathbf{w}^T T_i^\eta(\mathbf{f}_k) + b) = 0$, then (8) is not differentiable. The subgradient of a function $h : \mathbb{R}^N \rightarrow \mathbb{R}$ given at a point $\mathbf{f} \in \mathbb{R}^N$ is defined as the set

$$\partial h(\mathbf{f}) = \{\mathbf{u} : h(\mathbf{z}) \geq h(\mathbf{f}) + \mathbf{u}^T(\mathbf{z} - \mathbf{f})\} \quad (\mathbf{u}, \mathbf{z} \in \mathbb{R}^N).$$

For convex and differentiable functions h , the subgradient coincides with the gradient. Furthermore, the subdifferentiable convex function h has a minimum at a given point $\mathbf{f}$ if and only if $0 \in \partial h(\mathbf{f})$. This latter property allows for the construction of stochastic gradient methods which make use of

subgradients at points, where the objective function is not differentiable. This approach is not without precedent in machine learning. In fact, most modern backpropagation (and hence gradient-based) training methods are frequently used by models containing not everywhere differentiable layers (for example, ReLU activation functions in Ref. 32).

We now present the simplified objectives and gradients needed to train a VP-SVM classifier using stochastic gradient descent.^{20,27} In each step of the SGD algorithm, we randomly select a training example $\mathbf{f}_k$ ($k = 1, \dots, q$), calculate the gradient of the objective with respect to the trainable parameters and update them. This requires the simplification of the VP-SVM objectives proposed in Sec. 2.2. When used with SGD, objective (8) becomes

$$J(\mathbf{w}, \boldsymbol{\eta})_k := q \cdot C \cdot \max(0, 1 - y_k(\mathbf{w}^T T_i^\eta(\mathbf{f}_k))) + \|\mathbf{w}\|_2^2 + R(\boldsymbol{\eta}, \mathbf{f}_k) \quad (15)$$

where T_i^η ($i \in \{0, 1, 2\}$) denotes the variable projection transformations introduced in Sec. 2.1 and $R(\boldsymbol{\eta}, \mathbf{f}_k)$ is a simplified regulatory term:

$$R(\boldsymbol{\eta}, \mathbf{f}_k) := \alpha \cdot \frac{\|\mathbf{f}_k - T_1^\eta(\mathbf{f}_k)\|_2^2}{\|\mathbf{f}_k\|_2^2} \quad (\alpha \in \mathbb{R}). \quad (16)$$

Now, we can formulate the (sub) gradients of (15) with respect to $\mathbf{w}$ and $\boldsymbol{\eta}$. Note that we omitted the bias parameter b from (15) for simplicity, but it can be easily re-introduced. The subgradients of (15) exist with respect to $\mathbf{w}$ and $\boldsymbol{\eta}$ and can be expressed as

$$\frac{\partial J(\mathbf{w}, \boldsymbol{\eta})_k}{\partial \mathbf{w}} = \begin{cases} \mathbf{w} - C \cdot q \cdot y_k \cdot T_i^\eta(\mathbf{f}_k) & s_k > 0 \\ \mathbf{w} & s_k \leq 0, \end{cases} \quad (17)$$

where $s_k := 1 - y_k(\mathbf{w}^T T_i^\eta(\mathbf{f}_k))$ ($i \in \{0, 1, 2\}$) and

$$\frac{\partial J(\mathbf{w}, \boldsymbol{\eta})_k}{\partial \boldsymbol{\eta}} = \begin{cases} -C \cdot q \cdot y_k \cdot \frac{\partial T_i^\eta(\mathbf{f}_k)}{\partial \boldsymbol{\eta}} + \frac{\partial R(\boldsymbol{\eta}, \mathbf{f}_k)}{\partial \boldsymbol{\eta}} & s_k > 0 \\ \frac{\partial R(\boldsymbol{\eta}, \mathbf{f}_k)}{\partial \boldsymbol{\eta}} & s_k \leq 0. \end{cases} \quad (18)$$

The partial derivatives of the transformation $T_i^\eta(\mathbf{f}_k)$ ($i \in \{0, 1, 2\}$) depend on the choice of the index i

(see also Ref. 17). Suppose first ($i = 0$), i.e. the orthogonal transformation is being used as a feature extraction and dimension reduction step according to (3). Then, the partial derivative with respect to $\boldsymbol{\eta}$ can be given as

$$\frac{\partial T_0^\eta(\mathbf{f}_k)}{\partial \boldsymbol{\eta}} = \frac{\partial \Phi(\boldsymbol{\eta})^+ \mathbf{f}_k}{\partial \boldsymbol{\eta}} = \frac{\partial \Phi(\boldsymbol{\eta})^+}{\partial \boldsymbol{\eta}} \mathbf{f}_k, \quad (19)$$

where

$$\begin{aligned} \frac{\partial \Phi(\boldsymbol{\eta})^+}{\partial \boldsymbol{\eta}} &= -\Phi(\boldsymbol{\eta})^+ \frac{\partial \Phi(\boldsymbol{\eta})}{\partial \boldsymbol{\eta}} \Phi(\boldsymbol{\eta})^+ \\ &\quad + \Phi(\boldsymbol{\eta})^+ [\Phi(\boldsymbol{\eta})^+]^T \frac{\partial \Phi(\boldsymbol{\eta})}{\partial \boldsymbol{\eta}} \\ &\quad \times (I - \Phi(\boldsymbol{\eta}) \Phi(\boldsymbol{\eta})^+) + (I - \Phi(\boldsymbol{\eta})^+ \Phi(\boldsymbol{\eta})) \\ &\quad \times \left[\frac{\partial \Phi(\boldsymbol{\eta})}{\partial \boldsymbol{\eta}} \right]^T [\Phi(\boldsymbol{\eta})^+]^T \Phi(\boldsymbol{\eta})^+. \end{aligned}$$

Now let us consider the case when ($i = 1$) and $T_1^\eta(\mathbf{f}_k) = \Phi(\boldsymbol{\eta}) \Phi(\boldsymbol{\eta})^+ \mathbf{f}_k$ as given in (4). In this case,

$$\frac{\partial T_1^\eta(\mathbf{f}_k)}{\partial \boldsymbol{\eta}} = \frac{\partial [\Phi(\boldsymbol{\eta}) \Phi(\boldsymbol{\eta})^+]}{\partial \boldsymbol{\eta}} \mathbf{f}_k, \quad (20)$$

where

$$\begin{aligned} \frac{\partial [\Phi(\boldsymbol{\eta}) \Phi(\boldsymbol{\eta})^+]}{\partial \boldsymbol{\eta}} &= (I - \Phi(\boldsymbol{\eta}) \Phi(\boldsymbol{\eta})^+) \frac{\partial \Phi(\boldsymbol{\eta})}{\partial \boldsymbol{\eta}} \Phi(\boldsymbol{\eta})^+ \\ &\quad + \left[(I - \Phi(\boldsymbol{\eta}) \Phi(\boldsymbol{\eta})^+) \frac{\partial \Phi(\boldsymbol{\eta})}{\partial \boldsymbol{\eta}} \Phi(\boldsymbol{\eta})^+ \right]^T. \end{aligned}$$

It is easy to see that if ($i = 2$), or in other words, if we choose the transformation $T_2^\eta(\mathbf{f}_k) = \mathbf{f}_k - \Phi(\boldsymbol{\eta}) \Phi(\boldsymbol{\eta})^+ \mathbf{f}_k$, then

$$\frac{\partial T_2^\eta(\mathbf{f}_k)}{\partial \boldsymbol{\eta}} = -\frac{\partial T_1^\eta(\mathbf{f}_k)}{\partial \boldsymbol{\eta}}, \quad (21)$$

where $\frac{\partial T_1^\eta(\mathbf{f}_k)}{\partial \boldsymbol{\eta}}$ is given in (20). The gradient of $R(\boldsymbol{\eta}, \mathbf{f}_k)$ with respect to $\boldsymbol{\eta}$ is given as

$$\begin{aligned} \frac{\partial R(\boldsymbol{\eta}, \mathbf{f}_k)}{\partial \boldsymbol{\eta}} &= -\frac{2\alpha}{\|\mathbf{f}_k\|_2^2} \mathbf{f}_k^T (I - \Phi(\boldsymbol{\eta}) \Phi(\boldsymbol{\eta})^+) \\ &\quad \times \frac{\partial \Phi(\boldsymbol{\eta})}{\partial \boldsymbol{\eta}} \Phi(\boldsymbol{\eta})^+ \mathbf{f}_k. \end{aligned} \quad (22)$$

Once the partial derivatives of (15) have been calculated with respect to $\mathbf{w}$ and $\boldsymbol{\eta}$, we can update the parameters by

$$\begin{aligned} \boldsymbol{\eta} &\rightarrow \boldsymbol{\eta} - \gamma_k \cdot \frac{\partial J(\mathbf{w}, \boldsymbol{\eta})}{\partial \boldsymbol{\eta}} \quad \text{and} \\ \mathbf{w} &\rightarrow \mathbf{w} - \gamma_k \cdot \frac{\partial J(\mathbf{w}, \boldsymbol{\eta})}{\partial \mathbf{w}}. \end{aligned} \quad (23)$$

The SGD algorithm guarantees convergence to a local minimum, provided that the learning rate γ_k in (23) is small enough. We note that, instead of SGD, any other gradient-based optimization method may be used for training. Such methods include popular momentum-based approaches such as the Adam³³ algorithm. Note that our pytorch³⁴ based example implementation²⁶ allows the use of algorithms other than SGD, however investigating the effect of the use of different optimization methods for training was outside the scope of this work.

Similar to (15), a simplified form of the nonlinear VP-SVM objective (14) can be obtained for use with the SGD algorithm:

$$\begin{aligned} J(\boldsymbol{\beta}, \boldsymbol{\eta})_k &:= R(\boldsymbol{\eta}, \mathbf{f}_k) + q \cdot C \cdot \max(0, 1 - y_k \\ &\quad \times \sum_{j=1}^q \beta_j \cdot k(T_i^\eta(\mathbf{f}_k), T_i^\eta(\mathbf{f}_j))), \end{aligned} \quad (24)$$

where $R(\boldsymbol{\eta}, \mathbf{f}_k)$ is defined by (16) and $k(\cdot, \cdot)$ denotes an appropriate kernel function. Similar to the linear case, the (sub) gradient of (24) exists everywhere and is given by

$$\frac{\partial J(\boldsymbol{\beta}, \boldsymbol{\eta})_k}{\partial \beta_j} = \begin{cases} -C \cdot q \cdot k(T_i^\eta(\mathbf{f}_k), T_i^\eta(\mathbf{f}_j)) & s_k > 0 \\ 0 & s_k \leq 0, \end{cases} \quad (25)$$

and

$$\frac{\partial J(\boldsymbol{\beta}, \boldsymbol{\eta})_k}{\partial \boldsymbol{\eta}} = \begin{cases} \frac{\partial R(\boldsymbol{\eta}, \mathbf{f}_k)}{\partial \boldsymbol{\eta}} - C q y_k \sum_{j=1}^q \beta_j \\ \quad \times \frac{\partial k(T_i^\eta(\mathbf{f}_k), T_i^\eta(\mathbf{f}_j))}{\partial \boldsymbol{\eta}} & s_k > 0 \\ \frac{\partial R(\boldsymbol{\eta}, \mathbf{f}_k)}{\partial \boldsymbol{\eta}} & s_k \leq 0, \end{cases} \quad (26)$$

where $s_k := 1 - \sum_{j=1}^q \beta_j k(T_i^\eta(\mathbf{f}_k), T_i^\eta(\mathbf{f}_j))$. Clearly, the term $\frac{\partial k(T_i^\eta(\mathbf{f}_k), T_i^\eta(\mathbf{f}_j))}{\partial \boldsymbol{\eta}}$ depends on the transformation T_i^η ($i \in \{0, 1, 2\}$) and the chosen kernel function $k(\cdot, \cdot)$. In our experiments detailed in Sec. 4, we used VP-SVM setups with RBF kernel functions:

$$k(\mathbf{f}, \mathbf{g}) := e^{-\frac{\|\mathbf{f}-\mathbf{g}\|_2^2}{\sigma^2}} \quad (\mathbf{f}, \mathbf{g} \in \mathbb{R}^N, \sigma \in \mathbb{R}). \quad (27)$$

Specifically, for this kernel function, we can write the partial derivative with respect to η as

$$\begin{aligned} & \frac{\partial k(T_i^\eta(\mathbf{f}_k), T_i^\eta(\mathbf{f}_j))}{\partial \eta} \\ &= -\frac{2}{\sigma^2} \cdot e^{\frac{-\|T_i^\eta(\mathbf{f}_k) - T_i^\eta(\mathbf{f}_j)\|_2^2}{\sigma^2}} \\ & \cdot (T_i^\eta(\mathbf{f}_k) - T_i^\eta(\mathbf{f}_j)) \cdot \left[\frac{\partial T_i^\eta(\mathbf{f}_k)}{\partial \eta} - \frac{\partial T_i^\eta(\mathbf{f}_j)}{\partial \eta} \right]. \end{aligned}$$

Note that in the simplified nonlinear VP-SVM objective (24), we disregarded the regularization term $\sum_{k,j=1}^q \beta_k \beta_j k(T_i^\eta(\mathbf{f}_k), T_i^\eta(\mathbf{f}_j))$ of the full objective (14). This omission was done to minimize the computational cost of a single evaluation. Re-adding this term would require an $\mathcal{O}(q^2)$ expensive calculation after each update of the parameter vector η , which would significantly slow down the training process. Fortunately, as confirmed by our experiments in Sec. 4, the regulatory term (16) addresses the problem of vanishing gradients adequately and seems sufficient to produce well performing VP-SVM models for real-world classification problems. We note that even though in this work we focus on binary classification tasks, we see no theoretical barrier for the application of multiclass²⁵ SVM strategies for the proposed VP-SVM classifier.

2.4. Remarks on the implementation

We note that several effective training approaches exist for SVM objectives²⁷ that differ from the classical SGD approach. These include fast, packing-based algorithms for the primal objectives.³⁵ Investigating the possibility of adapting these algorithms to be used with VP-SVM is part of our future plans. However, the SGD approach does have important advantages from an implementation point of view. Namely, minimizing the simplified VP-SVM objective (24) using SGD allows us to “split” the proposed classifier into a variable projection “layer” and an SVM “layer”. This representation allows for the use of backpropagation techniques,³⁶ which further speed up the calculation of the gradients discussed in the previous section. Figure 1 demonstrates an inference step of the proposed VP-SVM classifier.

In order to demonstrate the results in this paper and support reproducible research, we created an example implementation of the proposed methods.²⁶

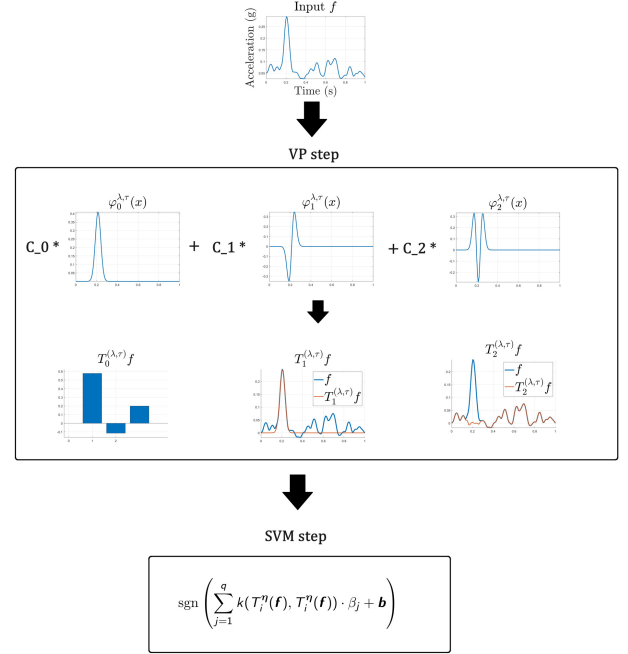


Fig. 1. (Color online) The progression of the data during a single inference step of VP-SVM. The input signals are represented by one of the T_i^η transformations. In this example, so-called adaptive Hermite functions were used (see Sec. 3.1), therefore $\eta = (\lambda, \tau)$ correspond to a dilation and translation parameter. The transformed signals are then passed to the underlying SVM classifier. Parameters of the VP-transformation and the weights of the classifier are optimized together during training.

Our implementation uses the pytorch³⁴ library, which allows for flexible parametrization and supports future investigations of the method. All of the VP-SVM related results discussed in Sec. 4 were acquired using this implementation.

3. Hermite Expansions

In this section, we discuss adaptive Hermite functions, which we used throughout our applications which we used in all of our VP-SVM-based applications (see Sec. 4). Hermite functions, which are closely related to the classical orthogonal Hermite polynomials have long been used to model 1D signals in a variety of fields. For example, their favorable morphology (see Fig. 2) and quickly decreasing modulus makes them appropriate to approximate ECG, EEG and other types of biomedical signals.^{6,19,21,37} In addition, Hermite expansions and

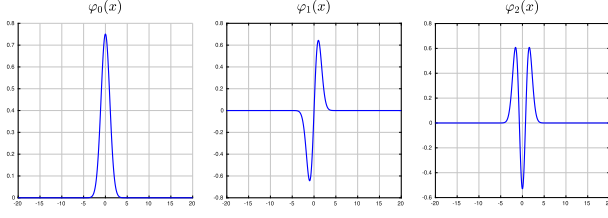


Fig. 2. (Color online) The first three Hermite functions. They can be used to efficiently approximate quasi-compactly supported signals from $L_2(\mathbb{R})$.

their generalizations have been successfully used to solve problems in engineering.¹⁸

We begin by discussing a generalization of Hermite functions known as adaptive Hermite functions,²¹ then we consider the discrete orthogonal variants of these functions which can produce more precise approximations of expansion coefficients while reducing computational costs when used with VP-SVM.

3.1. Adaptive Hermite functions

Consider the classical Hermite polynomials denoted henceforth by h_k , where $k \in \mathbb{N}$ is the order of the polynomial. These form a complete system in the weighted Lebesgue space $L_{2,w}(\mathbb{R})$ and are orthogonal with respect to the inner product

$$\langle h_k, h_j \rangle = \int_{-\infty}^{\infty} h_k(x) h_j(x) w(x) dx = \delta_{k,j} 2^k k! \sqrt{\pi},$$

where $(k, j \in \mathbb{N})$. The positive measure $w(x) = e^{-x^2}$ is also referred to as the weight function corresponding to the orthogonal Hermite polynomials.^{38,39} Hermite polynomials inherit a number of properties usual for orthogonal polynomials, such as adhering to the recurrence formulas:

$$\begin{aligned} h_k(x) &= 2x \cdot h_{k-1}(x) - 2(k-1) \cdot h_{k-2}(x) \\ h_0(x) &= 1, \quad h_1(x) = 2x \quad (k = 2, 3, 4, \dots), \end{aligned} \quad (28)$$

and

$$h'_k(x) = 2k \cdot h_{k-1}(x). \quad (29)$$

These polynomials have exactly k real roots, which are symmetric to zero and satisfy the interlacing property.³⁸ Using Hermite polynomials, the complete and orthonormal system in $L_2(\mathbb{R})$ known as Hermite functions can be defined:

$$\varphi_k(x) := h_k(x) \cdot e^{-x^2/2} / \sqrt{2^k k! \pi^{1/2}} \quad (k \in \mathbb{N}). \quad (30)$$

In (30), h_k denotes the k th Hermite polynomial. Hermite functions display a number of interesting and useful properties that make them important for applications and theory alike. For example, from a theoretical point of view one of the most well-known properties of Hermite functions is that they can be interpreted as eigenfunctions of the Fourier transform.⁴⁰ In case of applications, Hermite functions are often used because the first few functions display similar waveforms to many naturally appearing biomedical signals (such as ECG or EEG)^{19,21,41,37} and $|\varphi_k(x)|$ tends to zero quickly as $|x|$ increases for all $k \in \mathbb{N}$. These properties allow for the high quality approximation of “quasi” compactly supported signals $f \in L_2(\mathbb{R})$ with Hermite–Fourier partial sums using only the first few Hermite functions:

$$f \approx \sum_{k=0}^{n-1} \langle f, \varphi_k \rangle \varphi_k \quad (n \in \mathbb{N}).$$

The quantities $\langle f, \varphi_k \rangle$ are known as Hermite–Fourier coefficients and can be calculated using the usual inner product defined in $L_2(\mathbb{R})$:

$$\langle f, g \rangle = \int_{-\infty}^{\infty} f(x) \bar{g}(x) dx. \quad (31)$$

The first three Hermite functions are illustrated in Fig. 2.

In Ref. 21, the affine argument transforms of (30) were considered to model ECG signals:

$$\begin{aligned} \varphi_k^{\lambda, \tau}(x) &:= \sqrt{\lambda} \varphi_k(\lambda(x - \tau)) \\ (\tau, x &\in \mathbb{R}, \lambda > 0). \end{aligned} \quad (32)$$

These so-called adaptive Hermite functions also form a complete and orthonormal basis in $L_2(\mathbb{R})$ and retain the above-mentioned useful properties of Hermite functions. Furthermore, adaptive Hermite functions and their derivatives can also be easily calculated with the help of (28) and (29). For this reason, we can define adaptive orthogonal projection operators discussed in Sec. 2.1 using adaptive Hermite functions. In this case, the free parameters of the transformation are the dilation λ , which represents the magnitude of the unit step and the translation τ which represents a shift in time. More precisely, when using adaptive Hermite functions to define a variable projection operator, the parameter vector $\boldsymbol{\eta}$ used in the preceding sections can be given as $\boldsymbol{\eta} := [\lambda, \tau]^T \subset \mathbb{R}^2$.

These parameters (if applied to a suitable problem) often have physical interpretations. For example, in Ref. 18 we used adaptive Hermite functions to represent signals produced by a force sensor implanted into the tyre of a vehicle. In Ref. 18, it was shown for example that the optimal dilation parameter λ represents the speed of the vehicle. We use adaptive Hermite function-based VP-SVM classifiers to solve two real-world problems in Sec. 4.

In each of these applications, we will assume that the input examples are “quasi-periods” of an underlying 1D time series, and are “quasi-compactly” supported. Denote the effective support of the signal $f : \mathbb{R} \rightarrow \mathbb{R}$ by $\text{esupp} := \{x : |f(x)| > \varepsilon \geq 0\}$. To provide a more precise definition for the above terms, assume that the input examples are signals, whose effective support can be divided into $L \in \mathbb{N}$ intervals:

$$\text{esupp}(f) = \cup_{k=1}^L I_k, \quad \cap_{k=1}^L I_k = \emptyset.$$

We refer to the function values over these I_k intervals as “quasi-periods” of the signal and assume that there exists $C_k \geq 0$ such that

$$|f(x)| < C_k \cdot e^{-(x-t_k)^2} \quad (x \in I_k, \quad k = 1, \dots, L)$$

holds, where t_k denotes the midpoint of the interval I_k . Thus, the terms “quasi-periodic” and “quasi-compactly supported” mean that the effective support of the signal can be divided into distinct intervals on which signal values tend to zero quickly in either direction.

In this work, we assume that the time series f has already been segmented into these “quasi-periods” before being modeled by the adaptive Hermite system. In many applications (such as the ones discussed in Sec. 4) these are not very strict assumptions, because the quasi-compact support is usually guaranteed by the measurements’ natural morphology and often the segmentation is made easier by the underlying physical processes that generate the signals. Finally, we note that further generalizations of Hermite functions⁶ have also been introduced and used to successfully solve several signal processing problems.

3.2. Discrete adaptive Hermite functions

According to Sec. 2.1, we can define the adaptive Hermite function-based orthogonal projection used

by VP-SVM as

$$T_0^{\lambda, \tau}(\mathbf{f}) := \Phi(\lambda, \tau)^+ \mathbf{f} = \mathbf{c} \in \mathbb{R}^n, \quad (33)$$

where $\mathbf{f} \in \mathbb{R}^N$ is an input data example, $\Phi(\lambda, \tau) \in \mathbb{R}^{N \times n}$ is a matrix, whose columns are made up of discrete samplings of adaptive Hermite functions and $\lambda > 0, \tau \in \mathbb{R}$ are the free dilation and translation parameters given in (32). In this section, we consider type (3) transformations only, but the calculations are analogous to the other two types of transformations in (4) and (5). Suppose that the j th component of the k th column of $\Phi(\lambda, \tau)$ can be written as

$$\begin{aligned} \Phi(\lambda, \tau)_{jk} &= \varphi_k^{\lambda, \tau}(x_j) \\ (k &= 0, \dots, n-1, \quad j = 1, \dots, N), \end{aligned}$$

where the nodes x_j denote an equidistant sampling of the interval $I := [a, b]$, ($a, b \in \mathbb{R}$) and I contains the effective support of $\varphi_k^{\lambda, \tau}$ ($k = 0, \dots, n-1$). Then, the components of the coefficient vector $\mathbf{c}$ from (33) can be interpreted as a numerical approximation of the integral given in (31) using Riemann sums.

We will now propose the use of discrete orthogonal Hermite functions instead of the above construction when evaluating the transformation (33). The use of discrete orthogonal function systems is beneficial for the following reasons:

- For the discrete orthogonal adaptive Hermite system, in (33), $\Phi(\lambda, \tau)^+ = \Phi(\lambda, \tau)^T$ holds, which significantly speeds up the calculation of the pseudo inverse. The latter usually involves finding the singular value decomposition of the matrix $\Phi(\lambda, \tau)$, which is the most computationally expensive step in the evaluation of (33).
- The numerical quadrature used to evaluate the coefficients $\mathbf{c}$ becomes a Gaussian quadrature.³⁹ Namely,

$$\mathbf{c}_k = \int_{-\infty}^{\infty} f(x) \overline{\varphi_k^{\lambda, \tau}}(x) dx \quad (k = 0, \dots, n-1)$$

will hold for any signal $f(x) = P(x) \cdot e^{-x^2/2}$, where P can be an arbitrary polynomial of at most $2N-1$ degree.

Suppose that $N \geq 1$ is an arbitrary integer. It is well known that^{42,43}

$$\sum_{j=1}^{N+1} w_j h_i(x_j) h_s(x_j) = \delta_{is} \cdot 2^i i! \sqrt{\pi}, \quad (34)$$

where h_i and h_s ($i, s = 0, \dots, N-1$) denote the classical Hermite polynomials and the j th discrete positive weight value is given by

$$w_j = \frac{2^N N! \cdot \sqrt{\pi}}{(N+1) \cdot h_N^2(x_j)}. \quad (35)$$

Furthermore, the nodes satisfy $h_{N+1}(x_j) = 0$ ($j = 1, \dots, N+1$).^{39,42} From Eqs. (34) and (35), we can construct the discrete orthonormal Hermite functions

$$\varphi_{k,j} := h_k(x_j) \cdot \sqrt{w_j} / C_k \quad (j = 1, \dots, N), \quad (36)$$

where $C_k = \sqrt{2^k k! \pi^{1/2}}$, and so

$$\|\varphi_k\|_2^2 = \sum_{j=1}^N \varphi_{k,j}^2 = 1 \quad (k = 0, \dots, N-1)$$

is satisfied. By (34), the vectors φ_k clearly also satisfy the discrete orthogonal property

$$\langle \varphi_k, \varphi_s \rangle = \delta_{ks}.$$

Consider now some fixed $\lambda > 0$ and $\tau \in \mathbb{R}$ parameters. Then, zeros of the $(N+1)$ st degree adaptive Hermite function can be expressed as

$$\tilde{x}_j = x_j / \lambda + \tau \quad (j = 1, \dots, N+1), \quad (37)$$

where x_j denote the zeros of the $(N+1)$ st Hermite function. Consider the modified discrete weights

$$\tilde{w}_j = \frac{1}{(N+1) \cdot h_N^{\lambda, \tau}(\tilde{x}_j)},$$

where $h_N^{\lambda, \tau}(x) = h_N(\lambda(x - \tau))$. Constructing the vectors

$$\tilde{\varphi}_{k,j} := h_k^{\lambda, \tau}(\tilde{x}_j) \cdot \sqrt{\tilde{w}_j} \quad (k = 0, \dots, N-1), \quad (38)$$

we get an orthogonal system:

$$\langle \tilde{\varphi}_k, \tilde{\varphi}_s \rangle = \delta_{ks} \cdot C_k.$$

The constants C_k may easily be determined by taking the values of the diagonal matrix $(\tilde{\Phi}^{\lambda, \tau})^T \tilde{\Phi}^{\lambda, \tau}$, where the k th column of the matrix $\tilde{\Phi}^{\lambda, \tau}$ consists of the vector $\tilde{\varphi}_k$. Then, we can express the k th discrete orthonormal adaptive Hermite function by

$$\varphi_k^{\lambda, \tau} := h_k^{\lambda, \tau}(\tilde{x}_j) \cdot \sqrt{\tilde{w}_j} / \sqrt{C_k} \quad (k = 0, \dots, N-1, j = 1, \dots, N). \quad (39)$$

Note that several fast algorithms exist to calculate the nodes x_j (and $\tilde{x}_j$ due to (37)).³⁹ In practice, the above described discrete orthogonal functions would be most beneficial (see the points at the

beginning of this section) when they are used to implement the transformation step of an already trained VP-SVM classifier. We note that the discrete orthogonal adaptive Hermite functions (39) evaluated over the nodes $\tilde{x}_j$ are equal to the discrete orthogonal Hermite functions (36) evaluated over x_j . For this reason, it suffices to calculate the discrete orthogonal Hermite functions a single time. To include the effect of dilation and translation into the transformation (33), consider the definition of the Hermite–Fourier coefficients for the continuous case:

$$\begin{aligned} c_k &= \int_{-\infty}^{\infty} f(x) \sqrt{\lambda} \cdot \varphi_k(\lambda(x - \tau)) dx \\ &= \int_{-\infty}^{\infty} f(x/\lambda + \tau) / \sqrt{\lambda} \cdot \varphi_k(u) du. \end{aligned}$$

Using the above change of variable, we propose the following quick algorithm for evaluating a VP-SVM with discrete orthogonal Hermite basis functions:

- Suppose a VP-SVM classifier has already been trained and optimal λ and τ parameters have been identified for transformation (33).
- Calculate the nodes x_j , the weights w_j from (35) and the corresponding discrete orthogonal Hermite functions.
- Resample the data examples $\mathbf{f}_k$ ($k = 1, \dots, q$) at the discrete adaptive Hermite nodes (37) and multiply them by $1/\sqrt{\lambda}$.

In transformation (33), the pseudo inverse Φ^+ can now be replaced by Φ^T and the resulting numerical quadrature is of Gaussian type. We highlight that step (2) needs to be done only once, after the training phase of the classifier is finished. Since the data examples $\mathbf{f}_k$ are assumed to have quasi-compact supports, the resampling step in (3) can in many cases be done relatively safely using interpolation techniques.

4. Applications and Experiments

In this section, we detail two real-world applications of VP-SVM, which highlight the benefits of the proposed classification scheme. In our first example, we use VP-SVM to classify abnormal peaks in an accelerometer measurement. This example will be used to highlight how the parameters of the proposed Hermite function-based variable projection transformations can be used to explain classification results.

Then, we test the effectiveness of the proposed VP-SVM models on a biomedical signal processing problem. We use the benchmark MIT-BIH arrhythmia database²² to recognize so-called ventricular ectopic beats (VEBs) in real ECG data. We compare the proposed model to the state of the art.²³ In addition, we show that the proposed model outperforms similar, neural network-based methods¹⁷ while still providing the benefits usually associated with SVM classifiers.

4.1. Sensor fault detection

Detecting the abnormal behavior of sensors is an important engineering problem. This is especially true in the automotive industry, where new vehicle designs have to pass an extensive testing phase to validate their dynamic behavior. These tests are usually performed by equipping the vehicle with different types of sensors and performing predefined maneuvers in a controlled environment. Unfortunately, due to hardware failure, artificial peaks can appear in the measurements. Because such abnormal peaks can influence the outcome of the vehicle testing process, it is important to recognize and eliminate them. This can however be a difficult signal processing problem. In the case of accelerometers, for example large amplitude, seemingly out-of-context peaks (a usual characteristic of the waveforms caused by faulty hardware) can naturally appear in the measured vibration signals as a result of so-called shock events.

In a previous work,⁴⁴ Deuschle *et al.* proposed a method based on the Dynamic Time Warping (DTW) algorithm, which is able to recognize peaks in accelerometer data that appeared as a result of hardware malfunction. We used this method to label the peaks of a real-world accelerometer measurement taken from a vehicle which we used as our ground truth data.

We will now proceed to show that using this ground truth data we can train a VP-SVM classifier which can also identify abnormal peaks with perfect accuracy and discuss how the output of the variable projection transformation can be interpreted. Even though on the data available to us, both the DTW-based method and VP-SVM can completely solve this anomaly recognition problem, in practice the

two methods can complement each other. While the DTW-based algorithm⁴⁴ is not a supervised learning method and therefore does not require training data, it may have a higher computational cost than VP-SVM. For this reason, using a pre-trained VP-SVM approach for real-time applications can be beneficial.

The training set was created from the accelerometer measurement illustrated in Fig. 3. The signal was measured by a low-cost accelerometer which was mounted on the wheel hub of the vehicle.⁴⁵ Peaks below a predefined amplitude threshold of $1\text{ g} = 9.8\text{ m/s}^2$ were disregarded. The peaks, whose amplitudes were greater than the above-mentioned threshold were extended to 1D signal segments, by taking their 62.5 ms neighborhood on either side. The training set for VP-SVM was then created from these data segments. Figure 4 illustrates two data

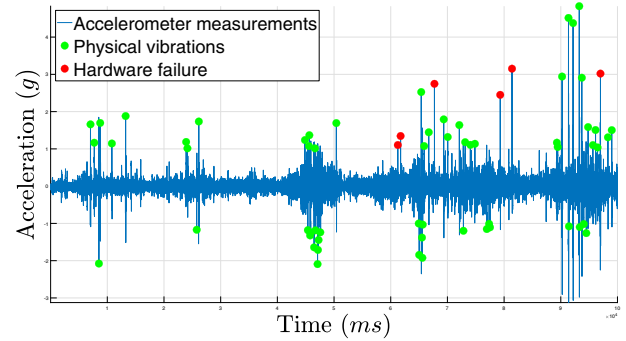


Fig. 3. (Color online) Peaks to be classified in the raw acceleration data. The green markings indicate that the peak was caused by a sudden physical impact, while red markings indicate peaks which were caused by hardware malfunction.

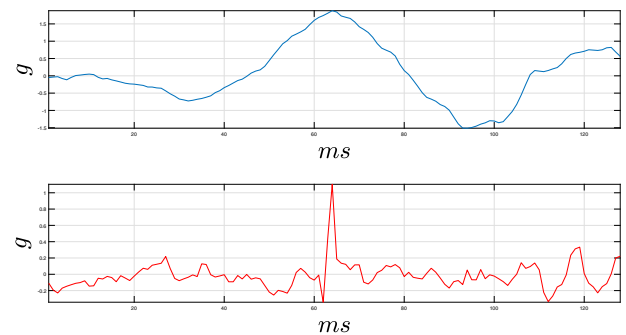


Fig. 4. (Color online) Data examples to be classified. Top: A peak corresponding to an actual physical event. Bottom: A peak appearing in the measurement due to hardware malfunction.

examples extracted from the measurement using the above steps.

As Fig. 3 shows, the dataset was small and rather ill-balanced: out of 65 peaks, only 6, ($\approx 9\%$ of the data) were caused by hardware issues. To overcome this, 4 abnormal data examples were selected randomly for the training set along with 4 randomly chosen normal examples. The rest of the examples (2 abnormal and 55 normal) peaks were added to the test set. As explained in detail below, after appropriate preprocessing steps, VP-SVM was able to completely separate the normal and abnormal examples in the training and test sets achieving 100 % accuracy on both.

A visual examination of the data segments shows that “abnormal” peaks usually constitute a single quick elevation with no decaying oscillations following it that could indicate the presence of an underlying physical process. The problem is made more difficult by abnormal examples showcased in Fig. 5, where abnormal peaks appear very close to, or on top of naturally occurring peaks. Running the above-detailed experiment by passing the raw data segments to an SVM or VP-SVM classifier yielded unsatisfactory results. Even though most of the time the classifiers could separate the examples in the training sets, abnormal examples in the test set were frequently mislabeled.

For this reason, a static feature extraction step was applied to the data examples before further attempts at classification. Since abnormal peaks seemed to appear in the data as single quick changes in signal value, we decided to take a time-frequency

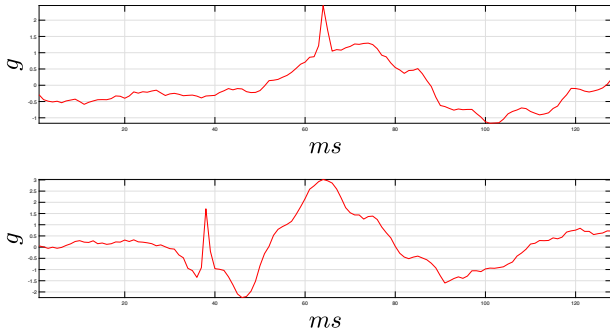


Fig. 5. (Color online) Difficult to identify abnormal data examples. Top: An abnormal peak appearing on top of a normal peak. Bottom: An abnormal peak appearing very close to a larger, normal peak.

representation, namely, the continuous wavelet coefficients of the data examples. The continuous wavelet transform⁴ was chosen for its redundant (and thus high resolution) representation of the data along the time axis and its translation invariant property. The latter property was especially useful, since the abnormal peaks could appear at different time instances in the data segments (see Fig. 5).

MatLab’s “cwt” method was used to calculate the wavelet coefficients using Morse analyzing wavelet.⁴⁶ The presence of abnormal peaks was indicated by well localized (in time), high absolute value wavelet coefficients corresponding to low scales (high frequencies). For this reason, instead of the complex wavelet coefficients, we considered the scalogram and disregarded coefficients corresponding to scales describing high frequencies. More precisely, if we consider a function $f \in L_2(\mathbb{R})$, we can formalize the above-mentioned transformation by

$$S_\psi(f; R, M, N) := \{|C_\psi^f(a_i, b_j)| : a_i > 0, b_j \in \mathbf{R}, (i = R, \dots, M; j = 1, \dots, L; R, M, N \in \mathbb{N})\}, \quad (40)$$

where

$$C_\psi^f(a, b) = \int_{-\infty}^{\infty} f(t) a^{-1/2} \overline{\psi}(a(t-b)) dt.$$

Clearly, the generated truncated scalogram $S_\psi(f; R, M, N)$ belongs to $\mathbb{R}^{M-R \times L}$. Choosing $R = 1$ in (40) yields the scalogram in the usual sense with respect to the wavelet ψ . Truncating the scalogram by choosing a large R means that $S_\psi(f)$ will contain the wavelet coefficients corresponding to scales representing high frequencies, which describe the quickly changing parts of the signal f . Thus, by choosing a large value for R , we performed a high pass filtering step. In our experiments, the scales a_i and the translations b_j were left as the default values from MatLab’s “cwt” routine.

Even though abnormal peaks seemed to separate well from the normal data examples based on the above described truncated scalogram representation, standard SVM classification still could not achieve perfect accuracy on the test set. In order to remedy this, a final transformation was applied to the truncated scalograms, by summing their values along the scale axis. Formally, the transformed data examples

$\tilde{\mathbf{f}}_k$ ($k = 1, \dots, q$) can be given as

$$(\tilde{\mathbf{f}}_k)_j = \sum_{i=R}^M |C_{\psi}^k(a_i, b_j)| \quad (j = 1, \dots, N),$$

where $\mathbf{f}_k \in \mathbb{R}^N$ denotes the k th data segment extracted from the accelerometer measurement. The transformed $\tilde{\mathbf{f}}_k \in \mathbb{R}^N$ signals and the corresponding scalograms are illustrated in Fig. 6.

As can be seen in Fig. 6, the transformed data examples $\tilde{\mathbf{f}}_k$ appear to be quasi-compact with a single Gaussian-like waveform, if the original peak was caused by hardware malfunction. If the original peak was caused by normal physical phenomena, the corresponding transformed signal segment usually displays more oscillations.

Finally, the transformed segments were used to train a VP-SVM classifier equipped with transformation (3) using the first 3 adaptive Hermite functions. The output of this transformation (after the VP-SVM classifier has been trained) captures the above-mentioned behavior of the input signals.

As shown in Fig. 7, the transformed abnormal acceleration signal segments were very well separated from the naturally occurring ones. This is to be expected, because if we analyze the waveforms of the transformed acceleration data examples (bottom row of Fig. 6), it is easy to see that signals representing hardware malfunction usually consisted of a single dominant peak and displayed a quasi-compact support. These attributes allowed for the good quality

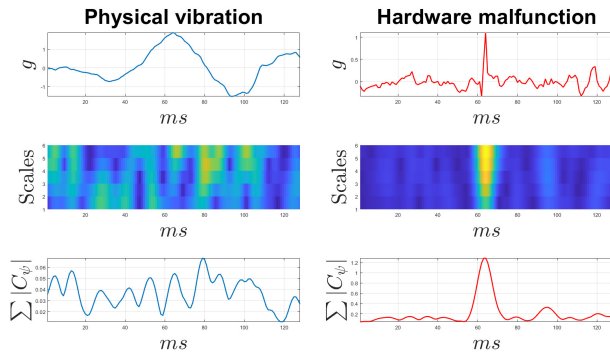


Fig. 6. (Color online) Top row: A normal and an abnormal data segment extracted from the original measurement. Middle row: Truncated Morse wavelet-based scalograms corresponding to the signals in the top row. Bottom row: 1D signals created by summing scalogram values along the time axis.

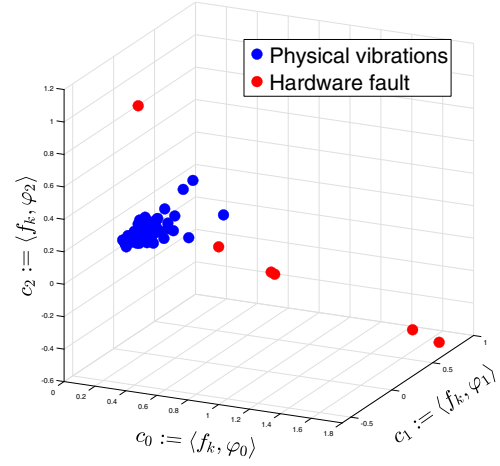


Fig. 7. (Color online) The optimal linear coefficients of the adaptive Hermite expansion for each data example (the output of transformation (3)). The parameters λ and τ were learned together with the weights of the underlying SVM classifier.

approximation of the transformed abnormal data segments by the first 3 adaptive Hermite functions, thus the Hermite–Fourier coefficients (31) differed significantly from 0. In contrast, the transformed data examples representing naturally occurring peaks display quick oscillations and their values do not remain near zero outside of a compact interval. As a result, the adaptive Hermite coefficients corresponding to these data examples will cluster around the origin indicating a large approximation error.

Furthermore, we highlight that the CWT-based preprocessing step alone is not enough to achieve a perfect classification on the test set. However, the application of VP-SVM to the signals acquired with the above process improves the classification accuracy up to 100%.

To demonstrate the effectiveness of the proposed VP-SVM approach in the sensor fault detection problem, we compared its performance with several classical machine learning approaches. In an industrial vehicle testing environment, algorithms are expected to run in near real time and on low cost hardware. These requirements exclude most hardware capable of supporting deep learning approaches, therefore such methods were not considered in our experiments. We refer to Sec. 4.2 for comparison of VP-SVM's performance with state-of-the-art deep learning methods.

In our current experiment, VP-SVM's ability to find faulty peaks in the accelerometer measurements was compared with 5 classical machine learning classifiers. In particular, we considered a classical SVM with RBF kernel (see e.g. Ref. 47), a random forest classifier,⁴⁸ a decision tree model,⁴⁹ a naive Bayes classifier⁵⁰ and a classification algorithm based on the gradient boosting method.⁵¹ In each experiment, identical training and test sets were considered. The training and test sets were chosen according to the beginning of this section. Data examples consisted of the transformed accelerometer measurements (see bottom row of Fig. 6).

The results of our experiment are given in Table 1. In addition to the accuracy score (Acc) achieved on the test set, we also provide the sensitivity/precision (Se) and predictivity/recall (+P) scores for each classifier. These statistics can be given by

$$\begin{aligned} \text{Acc} &= \frac{\text{TP} + \text{TN}}{\text{P} + \text{N}} \cdot 100, \\ \text{Se} &= \frac{\text{TP}}{\text{TP} + \text{FN}} \cdot 100, \\ +\text{P} &= \frac{\text{TP}}{\text{TP} + \text{FP}} \cdot 100. \end{aligned} \quad (41)$$

In Eq. (41), the terms TP, FP, TN, FN, P and N denote the number of true and false positive, true and false negative and total positive and negative model predictions.

The large discrepancy between the different statistics in Table 1 is due to the highly unbalanced nature of our dataset. In fact, out of the 57 data samples, only 2 denoted hardware faults in the test data. Hence, achieving high accuracy scores may occur at the expense of failing to identify the two specific input examples of interest. Nevertheless, the

proposed VP-SVM demonstrated the capability to predict samples representing sensor malfunctions, in addition to normal cases.

To assess the effectiveness of the proposed VP-transformation, we maintained identical hyperparameters for both the VP-SVM and the classical SVM classifier. In Table 1, it is evident that without the proposed VP-transformation, the SVM classifier overfits the training data and is incapable of recognizing sensor faults in the test set.

Conventional classifiers in Table 1 were tested using their respective implementations from the scikit-learn machine learning library.⁵² In these cases, hyperparameter tuning involved a manual grid search, with subsequent selection of the optimal values for the final experiments. The best performing hyperparameters were kept for our final experiments. Simulation results in Table 1 can be reproduced by using our pytorch implementation that is publicly available at the gitLab repository²⁶ along with its documentation. Clearly, none of the examined conventional classification schemes was able to match the performance of VP-SVM in this experiment, although some of them (Random Forest and Gradient Boosting) managed to avoid predicting any false negatives.

In practice, sensor fault detection should run in real time using low cost and simple hardware. Both of these requirements are important from the vehicle manufacturers' point of view, as they decrease the cost of vehicle tests. Even though certain hardware solutions capable of running deep learning models exist (e.g. the Nvidia Jetson series⁵³), they are usually significantly more expensive than low capacity microcontrollers.

To verify that the proposed method can be used in this context, we implemented VP-SVM on an STM32 F401RE microcontroller (see Fig. 8). This is a low cost (~13 EUR at the time of writing) computational unit with significant limitations. Indeed, the microcontroller uses an ARM 32-bit Cortex M4 CPU with a maximum frequency of 84 Hz. Furthermore, the board has only 512kB flash memory. We developed a C/C++ implementation of the inference steps of the proposed VP-SVM classifier. Table 2 shows the average, lowest and highest inference times measured for the processing of the peaks in our acceleration data.

Table 1. Comparison of VP-SVM and classical ML methods for sensor fault detection on the test set.

Classifier	Acc	Se	+P
VP-SVM (RBF kernel)	100.00	100.00	100.00
SVM (RBF kernel)	96.49	0.00	0.00
Random Forest	98.25	100.00	50.00
Decision Tree	96.49	50.00	50.00
Naive Bayes	96.49	50.00	50.00
Gradient Boost	98.25	100.00	50.00

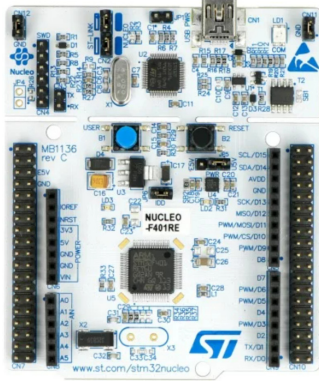


Fig. 8. (Color online) The STM32 F401RE microcontroller used in our experiments.

Table 2. Inference time of VP-SVM for sensor fault detection including the communication overhead between the microcontroller and the PC.

Average (s)	Highest (s)	Lowest (s)
0.3535	0.3523	0.3545

From Table 2, it is clear that the proposed VP-SVM classifier is suitable for near real-time sensor fault detection.

4.2. Arrhythmia detection

In this section, we evaluate the performance of the proposed VP-SVM in a standard classification benchmark problem. Such problems have long been handled by various ML approaches, as the relationship between medical signals and health conditions is usually highly nonlinear and cannot be modeled using traditional techniques.

In many applications, such as ECG and electroencephalography (EEG) classification, the presence of noise, the nonstationary behavior of the signals and the scarcity of annotated training data make the introduction of reliable ML models difficult. Such problems are mitigated by specific techniques, for example self-supervised learning schemes.⁵⁴

In this section, we show how the proposed method can be used to recognize a common heart irregularity. ECG signals are 1D time series that represent the electrical activity of the heart. These types of signals are one of the most important tools for diagnosing heart disease. A single heartbeat recorded from a

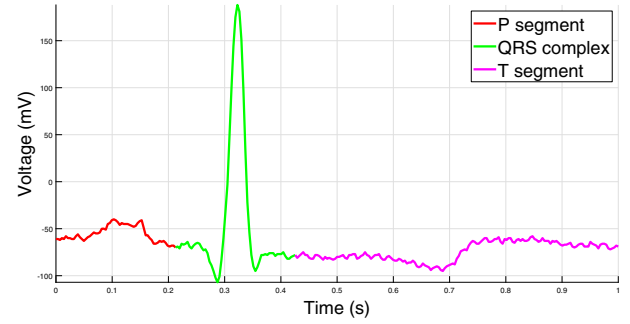


Fig. 9. (Color online) ECG representing a healthy heartbeat. The P wave, QRS complex and T wave correspond to atrial depolarization, ventricular depolarization and ventricular repolarization, respectively.

healthy subject is depicted in Fig. 9. An ECG segment describing a single heartbeat can be divided into several smaller segments that correspond to cardiac muscle depolarization and repolarization. Cardiologists can diagnose a multitude of heart function irregularities based on the shape, amplitude and temporal location of these segments.⁵⁵ In order to assist cardiologists with the evaluation of (long-term recordings, see e.g. Holter⁵⁶) ECG measurements and to develop automatic mobile heart monitoring applications,⁵⁷ many ECG processing methods have been proposed.²³

The MIT-BIH arrhythmia database²² is a standard benchmark for testing state-of-the-art methods. The original dataset provided by PhysioNet,⁵⁸ contains over 100,000 annotated heartbeats recorded from real patients. The database is highly unbalanced, with normal heartbeats (from healthy patients) being over represented. The heartbeats can be divided into several classes. Many works,²³ rely on some variation of the classes specified in the ANSI/AAMI EC57:1998 standard to categorize the heartbeats. Unfortunately, different works modify these classes slightly (for example grouping classes together) to counter the problem of the heavy bias towards normal heartbeats.

In this work, we focus on recognizing an important class of heartbeat irregularities known as ventricular ectopic beats (VEBs). There are multiple benefits to considering this subproblem. First, VEBs constitute the most common irregular heartbeat type in the MIT-BIH database, therefore the conducted experiments can be used to compare our proposed

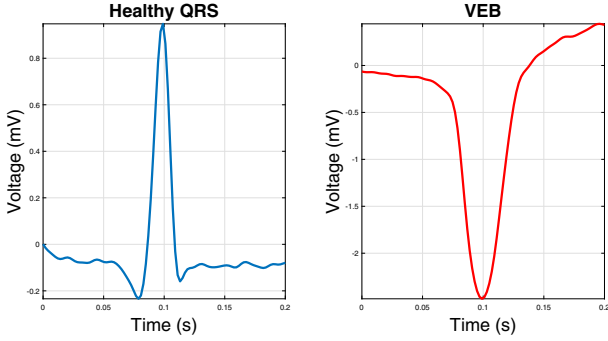


Fig. 10. (Color online) Left: A normal QRS complex. Right: A typical VEB waveform taken from the MIT-BIH arrhythmia database.

model's efficiency to the state of the art. Second, it is important to mention that in Ref. 17, a VP-enhanced model-driven neural network known as VP-NET was considered for the same problem. Our experiments can be used to fully compare the two models and draw conclusions about the benefits of the proposed VP-SVM classifier. Lastly, as Fig. 10 illustrates, VEBs often display an irregular QRS complex within the heartbeat, which are particularly well suited to be modeled by Hermite functions.^{6,21,59}

Current state-of-the-art approaches for the considered experiment include convolutional neural network-based methods⁶⁰ and traditional methods.^{61,62} In traditional approaches, a static feature extraction step is applied to the ECG signals before training. Among these methods, Hermite features proved to be a very efficient representation of QRS complexes due to their sparse nature,⁵⁹ discriminative power^{61–63} and interpretability.^{17,37} These results endorse our selection of Hermite functions for the representation and classification of QRS complexes.

In our experiments, the original dataset was split into a training and a test set according to Ref. 62, such that there is no data leakage, neither at the patient level nor at the recording level. In fact, QRS complexes in the training and tests sets come from different recordings of distinct patients. In our experiments, we utilized Ref. 64 to extract the QRS segment from each heartbeat as the input to the proposed model. We opted to use only the QRS segments, as these can be modeled with the first 8 Hermite functions.^{17,59} Since a number of previous methods^{17,64,65} also relied on the same transformation

to extract features, the use of QRS segments makes our results comparable to the state of the art.

In addition to the unbalanced dataset described above, we also tested our method on a balanced subset introduced in Ref. 17. This experiment allows for further comparison between the proposed VP-SVM classifier and the VP-NET model¹⁷ as well as classical neural network-based models. The balanced dataset contained every available VEB and the same amount of normal beats randomly sampled from the whole data. In total, 4260 normal heartbeats and 4260 VEBs were considered in the training set, while 3220 normal beats and 3220 VEBs were included in test set.

Tables 3 and 4 summarize the Acc, Se, and +P scores for the test set, which are usual metrics associated with evaluating ECG data.²³

Our results clearly demonstrate the efficiency of the proposed VP-SVM classification scheme in VEB detection. In Table 3, different classification architectures were compared using a balanced dataset. These experiments allowed for a fair comparison of the different classifiers. In addition to our own results on the dataset using the proposed VP-SVM classifier, Table 3 also showcases results achieved by competing models, such as a VP-NET, a fully connected neural network (FCNN) and a convolutional neural network (CNN). For each classifier, more than 3500 hyperparameter configurations were examined using grid search to achieve the above results. The VP-SVM and VP-NET classifiers were utilized with an initial VP layer that included the first $n = 8$ number of adaptive Hermite functions. Furthermore, the proposed VP-SVM classifier was equipped with the RBF kernel. This choice was appropriate not only because of its ability to deal with highly nonlinear classification problems, but also because RBF kernel

Table 3. ECG classification results on the test set. Models trained on balanced dataset (test set).

Classifier	Acc	Normal		VEB	
		Se	+P	Se	+P
VP-SVM	97.83	98.08	97.60	97.57	98.06
VP-NET	96.65	99.38	93.23	93.91	99.34
CNN	96.34	97.76	95.05	94.91	97.70
FCNN	94.38	93.79	94.91	94.97	93.86

SVMs have been previously successfully used for ECG classification.^{65,66} The learning rate, the parameter of the RBF kernel and the parameter α of the penalty term (9) were set by a grid search. In the hyperparameter optimization for VP-SVM, we examined 125 configurations, then the model that achieved the best accuracy on the validation set was chosen for the final evaluation on the test.

Table 3 shows that VP-SVM outperformed the other classifiers in terms of total accuracy on the test set. If we consider VEBs as “positive” samples, then VP-SVM achieves an Se score of 97.57%, the highest of the examined classification schemes. In fact, VP-SVM in this case outperforms the second best fully connected neural network model by 2.6% and the model-driven VP-NET architecture by 3.6%. From medical point of view Se is of utmost importance, since it indicates how well the proposed method can identify true positives, i.e. the presence of VEBs. In the balanced case, the proposed classifier is slightly outperformed (by 1.3% and 1.28%, respectively) only by the state-of-the-art VP-NET model in normal Se and VEB +P scores, however VP-SVM still achieves the best VEB Se score, achieving an over 2% increase compared to the next best classifier.

We draw similar conclusions from analyzing the results in Table 4. As stated above, this table presents results of a more realistic scenario, as VEBs were significantly underrepresented in both the training and test sets. We highlight that VP-SVM achieves an 11.24% better VEB Se score than the next best performing fully connected model, which is a significant improvement compared to the other classifiers. In addition to the examined models, we compared the Se and +P scores of VP-SVM to the

current state of the art. These methods include deep learning⁶⁰ and traditional methods (static feature extraction, then classification^{61,66,62}). Table 4 shows that the scores attained by the proposed VP-SVM approach range among the top-performing state-of-the-art ECG classifiers.

Besides its classification performance, VP-SVM offers additional advantages over state-of-the-art learning methods. Namely, the initial layer of the proposed model implements feature learning by using adaptive Hermite functions in the framework of variable projections. As discussed in Sec. 3.1, the learned parameters of VP transformations can often be interpreted in a physical sense. Indeed, in the examined ECG processing application, it can be shown that the learned dilation parameters correspond to the width of the QRS complexes in time (heartrate), while the translation parameter can be used to obtain the average location of the so-called R peak. These useful properties are shared by the VP-NET-based VEB detection algorithm introduced in Ref. 17, as well as traditional approaches employing VP transformations.⁶⁵ A key difference between these approaches however, is that the weights of the fully connected neural network following the VP transformation in VP-NET lack explainability. In contrast, the weights learned by VP-SVM can be identified with a hyper plane that separates the classes, thus they have a clear geometric interpretation. Another advantage of VP-SVM is that it is not as prone to overfitting as VP-NET (as explained in Ref. 17) and other, potentially deep neural networks. This is generally due to SVM-based classifiers’ resilience to overfitting,^{67,68} which is also supported by the results shown in Tables 3 and 4.

Table 4. ECG classification results on the test set. Models trained on the realistic unbalanced dataset (test set).

Classifier	Acc	Normal		VEB	
		Se	+P	Se	+P
VP-SVM	99.00	99.45	99.43	95.31	95.45
VP-NET	98.45	99.57	98.78	83.07	93.37
CNN	98.35	99.39	98.85	84.07	90.93
FCNN	97.49	98.50	98.81	83.70	80.21
state-of-the-art	–	80–99	85–99	77–96	63–99

5. Conclusion

In this paper, we further developed our novel classification scheme, which extends the classical SVM algorithm by adaptive orthogonal transformations. These transformations can be interpreted as an automatic feature extraction step applied to the data before we attempt to find the optimal hyperplane separating the classes. During training, the parameters of these feature extraction transformations are optimized alongside the weights

of the SVM classifier resulting in optimal data representation.

If the transformation is chosen correctly, the result of the feature extraction step and the classification scheme can be interpreted by humans. We discussed the objective functions, training methods and numerical optimization of the proposed classifier. To this end, we introduced discrete orthogonal adaptive Hermite functions and discussed how their use can further reduce the computational complexity of the proposed methods.

For the first time, we applied the proposed VP-SVM classification scheme to two, real-world engineering problems as discussed in Sec. 4. First, we applied the proposed classifier to recognize peaks in acceleration data caused by hardware malfunction. Then, we detected abnormal heartbeats in ECG data obtained from the benchmark dataset.²²

In the first application, we applied VP-SVM to the problem of finding peaks caused by hardware malfunction in a 1D time series. We showed that using the adaptive Hermite function-based transformation VP-SVM is able to distinguish between data segments containing the above mentioned abnormal peaks, and segments where the quick oscillation was caused by normal physical phenomena. To this end, we applied a predefined continuous wavelet transformation-based feature extraction step to the signals, then trained the above mentioned VP-SVM setup on the transformed training set. We discussed how the output of the trained transformation step of the VP-SVM can be interpreted by showing that the output of the learned orthogonal Hermite transformation confirms or disproves the presence of quick oscillations in the original measurements which are well localized in time.

We showed that the proposed VP-SVM outperforms traditional ML approaches for the sensor fault detection problem. In addition, we implemented VP-SVM on a low cost STM32 F401RE microcontroller and showed that the proposed method can achieve satisfactory inference times for near real-time use.

In the second application, we showed that the proposed VP-SVM classifier provides robust ability to recognize VEBs in ECG data. VP-SVM performed at the level of the best state-of-the-art methods both in terms of classification accuracy and other statistics. In fact, using accuracy, Se and +P scores we

showed that VP-SVM significantly outperforms similar model-driven neural networks, as well as other classification algorithms in the literature that can be considered state of the art. Importantly, VP-SVM is able to achieve this result while retaining the benefits associated with the use of SVM over deep learning methods. These benefits include better interpretability, being less prone to overfitting and simple model architecture.^{67,68}

We plan to continue our research by investigating the dual form of the VP-SVM objectives and the possibility of using nongradient-based training methods. Furthermore, we plan to optimize our low level implementations and conduct real-time experiments for vehicle testing. In addition, we plan to investigate and implement extensions of the proposed VP-SVM classifier for multiclass problems.

Another interesting future direction will be the investigation of classification problems, where inputs consist of 2D or higher dimensional inputs. Variable projection-based methods been previously adopted to 2D image processing problems including Bézier-curve fitting,⁶⁹ blind deconvolution,⁷⁰ and adaptive 2D spectrograms.⁷¹ Incorporating these results into a VP-SVM framework will be an important future objective of our research. We also plan to investigate surface-fitting problems by employing the tensor product of 1D free-knot splines.⁷²

Finally, we plan to investigate if the idea of introducing adaptive orthogonal representations of input vectors can be used together with the recently introduced ML architectures. In particular, we would like to investigate if dynamic ensemble methods such as Ref. 73 can be adopted to VP-SVM. Creating such ensemble models would allow for explainable ensemble ML models. This is of particular interest to us, as several ensemble approaches^{65,61} have been successfully applied for biomedical signal processing tasks similar to the problem investigated in Sec. 4.2.

Acknowledgments

Project no. C1748701 has been implemented with the support provided by the Ministry of Culture and Innovation of Hungary from the National Research, Development and Innovation Fund, financed under the NVKDP-2021 funding scheme. This project was

supported by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences. Project no. TKP2021-NVA-29 has been implemented with the support provided by the Ministry of Innovation and Technology of Hungary from the National Research, Development and Innovation Fund, financed under the TKP2021-NVA funding scheme.

Federico Deuschle was supported by a Marie Skłodowska Curie early stage researcher grant. We gratefully acknowledge the European Commission for its support of the Marie Skłodowska Curie program through the H2020 ETN MOIRA project (GA 955681).

We would like to thank Attila Miklós Ámon for his assistance with some of the numerical experiments discussed in this paper.

ORCID

Tamás Dózsa  <https://orcid.org/0000-0003-0919-4385>

Péter Kovács  <https://orcid.org/0000-0002-0772-9721>

References

1. J. Schmidhuber, Deep learning in neural networks: An overview, *Neural Netw.* **61** (2015) 85–117.
2. A. Krizhevsky, I. Sutskever and G. E. Hinton, ImageNet classification with deep convolutional neural networks, *Commun. ACM* **60**(6) (2017) 84–90.
3. H. Abdi and L. J. Williams, Principal component analysis, *WIREs Comput. Stat.* **2**(4) (2010) 433–459.
4. C. Gasquet and P. Witomski, *Fourier Analysis and Applications: Filtering, Numerical Computation, Wavelets* (Springer Science & Business Media, 2013).
5. P. Kovács, S. Fridli and F. Schipp, Generalized rational variable projection with application in ECG compression, *IEEE Trans. Signal Process.* **68** (2020) 478–492.
6. P. Kovács, C. Böck, T. Dózsa, J. Meier and M. Huemer, Waveform modeling by adaptive weighted hermite functions, *ICASSP 2019 - 2019 IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)* Brighton, UK, 2019, pp. 1080–1084.
7. G. Golub and V. Pereyra, Separable nonlinear least squares: the variable projection method and its applications, *Inverse Probl.* **19** (2003) R1.
8. J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai and T. Chen, Recent advances in convolutional neural networks, *Pattern Recogn.* **77** (2018) 354–377.
9. G. Montavon, W. Samek and K.-R. Müller, Methods for interpreting and understanding deep neural networks, *Digital Signal Process.* **73** (2018) 1–15.
10. M. H. Rafiei and H. Adeli, A new neural dynamic classification algorithm, *IEEE Trans. Neural Netw. Learn. Syst.* **28**(12) (2017) 3074–3083.
11. F. K. Doilovic, M. Brcic and N. Hlupic, Explainable artificial intelligence: A survey, *2018 41st Int. Convention Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, Opatija, Croatia, 2018, pp. 0210–0215.
12. J. R. Hershey, J. L. Roux and F. Weninger, Deep unfolding: Model-based inspiration of novel deep architectures (2014), <https://arxiv.org/abs/1409.2574>.
13. M. T. Ribeiro, S. Singh and C. Guestrin, Anchors: High-precision model-agnostic explanations, in *Proc. AAAI Conf. Artificial Intelligence*, Quebec, Canada, 2018.
14. K. Hammernik, T. Kstner, B. Yaman, Z. Huang, D. Rueckert, F. Knoll and M. Akakaya, Physics-driven deep learning for computational magnetic resonance imaging: Combining physics and machine learning for improved medical imaging, *IEEE Signal Process. Mag.* **40**(1) (2023) 98–114.
15. R. T. Q. Chen, Y. Rubanova, J. Bettencourt and D. K. Duvenaud, Neural ordinary differential equations, *Advances in Neural Information Processing Systems*, eds. S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi and R. Garnett, Vol. 31 (Curran Associates, 2018).
16. D. R. Pereira, M. A. Piteri, A. N. Souza, J. P. Papa and H. Adeli, FEMA: A finite element machine for fast learning, *Neural Comput. Appl.* **32** (2020) 6393–6404.
17. P. Kovács, G. Bognár, C. Huber and M. Huemer, VPNet: Variable projection networks, *Int. J. Neural Syst.* **32**(1) (2021) 2150054–1–19.
18. T. Dózsa, J. Radó, J. Volk, A. Kisari, A. Soumelidis and P. Kovács, Road abnormality detection using piezo-resistive force sensors and adaptive signal models, *IEEE Trans. Instrum. Measure.* **71** (2022) 1–11.
19. T. Dózsa, C. Beck, G. Bognár, J. Meier and P. Kovács, Color classification of visually evoked potentials by means of hermite functions, *2021 55th Asilomar Conf. Signals, Systems, and Computers*, Pacific Grove, CA, USA, 2021, pp. 251–255.
20. T. Dózsa and P. Kovács, Variable projection support vector machines, in *Proc. 4th Int. Conf. Advances in Signal Processing and Artificial Intelligence (ASPAI)*, Corfu, Greece, 2022, pp. 47–52.
21. T. Dózsa and P. Kovács, ECG signal compression using adaptive hermite functions, *ICT Innovations 2015* (Springer International Publishing, 2016), pp. 245–254.
22. G. B. Moody and R. G. Mark, The impact of the MIT-BIH arrhythmia database, *IEEE Eng. Med. Biology Mag.* **20**(3) (2001) 45–50.

23. E. J. D. S. Luz, W. R. Schwartz, G. Cámara-Chávez and D. Menotti, ECG-based heartbeat classification for arrhythmia detection: A survey, *Comput. Methods Prog. Biomed.* **127** (2016) 144–164.
24. G. H. Golub and V. Pereyra, The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate, *SIAM J. Numer. Anal.* **10**(2) (1973) 413–432.
25. K.-B. Duan and S. S. Keerthi, Which is the best multiclass SVM method? An empirical study, *Int. Workshop Multiple Classifier Systems*, Springer, Günzburg, Germany, 2005, pp. 278–285.
26. T. Dózsa, F. Desuschle, B. Cornelis and P. Kovács, Variable projection support vector machines - implementation, <https://gitlab.com/tamasdzs/vp-svm> (2023).
27. J. Shawe-Taylor and S. Sun, A review of optimization methodologies in support vector machines, *Neurocomputing* **74**(17) (2011) 3609–3618.
28. M. Grant and S. Boyd, Graph implementations for nonsmooth convex programs, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, eds. V. Blondel, S. Boyd and H. Kimura (Springer-Verlag, 2008), pp. 95–110.
29. M. Grant and S. Boyd, CVX: Matlab software for disciplined convex programming, version 2.1 (2014).
30. O. Chapelle, Training a support vector machine in the primal, *Neural Comput.* **19**(5) (2007) 1155–1178.
31. S. Shalev-Shwartz, Y. Singer and N. Srebro, Pegasos: Primal estimated sub-gradient solver for SVM, in *Proc. 24th Int. Conf. Machine Learning, ICML '07* (Association for Computing Machinery, 2007), pp. 807–814.
32. Y. Li and Y. Yuan, Convergence analysis of two-layer neural networks with RELU activation, *Advances in Neural Information Processing Systems*, eds. I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan and R. Garnett, Vol. 30 (Curran Associates, 2017).
33. D. P. Kingma and J. Ba, Adam: A method for stochastic optimization (2017), <https://arxiv.org/abs/1412.6980>.
34. A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai and S. Chintala, Pytorch: An imperative style, high-performance deep learning library (2019), arXiv:1912.01703.
35. A. Z. Zeyuan, C. Weizhu, W. Gang, Z. Chenguang and C. Zheng, P-packSVM: Parallel primal gradient descent kernel SVM, *2009 Ninth IEEE Int. Conf. Data Mining*, Miami, Florida, 2009, pp. 677–686.
36. B. J. Wythoff, Backpropagation neural networks: A tutorial, *Chemom. Intell. Lab. Syst.* **18**(2) (1993) 115–155.
37. M. Lagerholm, C. Peterson, G. Braccini, L. Edenbrandt and L. Sörnmo, Clustering ECG complexes using Hermite functions and self-organizing maps, *IEEE Trans. Biomed. Eng.* **47**(7) (2000) 838–848.
38. G. Szegő, *Orthogonal Polynomials* (American Mathematical Society, 1939).
39. W. Gautschi, *Orthogonal Polynomials: Computation and Approximation* (OUP Oxford, 2004).
40. G. G. Walter, Properties of hermite series estimation of probability density, *Ann. Stat.* **5**(6) (1977) 1258–1264.
41. C. Böck, P. Kovács, P. Laguna, J. Meier and M. Huemer, ECG beat representation and delineation by means of variable projection, *IEEE Trans. Biomed. Eng.* **68**(10) (2021) 2997–3008.
42. O. Streltsova and I. Streltsov, Hermite orthogonal polynomials of a discrete variable, *J. Comput. Methods Sci. Eng.* **2**(1–2) (2002) 237–244.
43. M. Abramowitz, I. A. Stegun and R. H. Romer, *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables* (University Libraries UNT Digital Library, 1988).
44. F. Deuschle, B. Cornelis and K. Gryllias, Robust sensor spike detection method based on dynamic time warping, in *Proc. 4th Int. Conf. Advances in Signal Processing and Artificial Intelligence (ASP AI)*, Corfu, Greece, 2022, pp. 20–27.
45. B. Cornelis and B. Peeters, Online bayesian spike removal algorithms for structural health monitoring of vehicle components, in *Proc. 9th Int. Conf. Structural Dynamics (EURODYN)* (Porto, Portugal, 2014), pp. 2295–2301.
46. J. M. Lilly and S. C. Olhede, Generalized morse wavelets as a superfamily of analytic wavelets, *IEEE Trans. Signal Process.* **60**(11) (2012) 6036–6041.
47. M. Ring and B. M. Eskofier, An approximation of the Gaussian RBF kernel for efficient classification with svms, *Pattern Recogn. Lett.* **84** (2016) 107–113.
48. A. Parmar, R. Katariya and V. Patel, A review on random forest: An ensemble classifier, *Int. Conf. intelligent Data Communication Technologies and Internet of things (ICICI) 2018* (Springer, Coimbatore, India, 2019), pp. 758–763.
49. S. B. Kotsiantis, Decision trees: A recent overview, *Artif. Intell. Rev.* **39** (2013) 261–283.
50. I. Rish *et al.*, An empirical study of the naive bayes classifier, *Int. Joint Conf. Artificial Intelligence, 2001 Workshop Empirical Methods in Artificial Intelligence*, Seattle, USA, 2001, pp. 41–46.
51. J. H. Friedman, Stochastic gradient boosting, *Comput. Stat. Data Anal.* **38**(4) (2002) 367–378.
52. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, Scikit-learn: Machine learning in Python, *J. Mach. Learn. Res.* **12** (2011) 2825–2830.
53. S. Cass, Nvidia makes it easy to embed AI: The jetson nano packs a lot of machine-learning power into

- diy projects - [hands on], *IEEE Spectr.* **57**(7) (2020) 14–16.
54. M. H. Rafiei, L. V. Gauthier, H. Adeli and D. Takabi, Self-supervised learning for electroencephalography, *IEEE Trans. Neural Netw. Learn. Syst.* (2022) 1–15.
55. R. Vecht, M. A. Gatzoulis and N. Peters, *ECG Diagnosis in Clinical Practice* (Springer Science & Business Media, 2009).
56. N. J. Holter, New method for heart studies: Continuous electrocardiography of active subjects over long periods is now practical, *Science* **134**(3486) (1961) 1214–1220.
57. P. Guzik and M. Malik, ECG by mobile technologies, *J. Electrocardiol.* **49**(6) (2016) 894–901.
58. A. L. Goldberger, L. A. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng and H. E. Stanley, Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals, *Circulation* **101**(23) (2000) e215–e220.
59. A. Sandryhaila, S. Saba, M. Puschel and J. Kovacevic, Efficient compression of QRS complexes using hermite expansion, *IEEE Trans. Signal Process.* **60**(2) (2011) 947–955.
60. S. Kiranyaz, T. Ince and M. Gabbouj, Real-time patient-specific ECG classification by 1d convolutional neural networks, *IEEE Trans. Biomed. Eng.* **63**(3) (2015) 664–675.
61. K. Park, B. Cho, D. Lee, S. Song, J. Lee, Y. Chee, I. Kim and S. Kim, Hierarchical support vector machine based heartbeat classification using higher order statistics and hermite basis function, *Computers in Cardiology*, Bologna, Italy, 2008, pp. 229–232.
62. P. De Chazal, M. O'Dwyer and R. B. Reilly, Automatic classification of heartbeats using ECG morphology and heartbeat interval features, *IEEE Trans. Biomed. Eng.* **51**(7) (2004) 1196–1206.
63. R. Jane, S. Olmos, P. Laguna and P. Caminal, Adaptive Hermite models for ECG data compression: Performance and evaluation with automatic wave detection, in *Proc. Int. Conf. Computers in Cardiology*, London, UK, 1993, pp. 389–392.
64. M. Sarfraz, A. A. Khan and F. F. Li, Using independent component analysis to obtain feature space for reliable ecg arrhythmia classification, *2014 IEEE Int. Conf. Bioinformatics and Biomedicine (BIBM)*, Belfast, UK, 2014, pp. 62–67.
65. T. Dózsa, G. Bognár and P. Kovács, Ensemble learning for heartbeat classification using adaptive orthogonal transformations, *Int. Conf. Computer Aided Systems Theory* (Springer, Las Palmas, Spain, 2019), pp. 355–363.
66. C. Ye, B. V. K. Vijaya Kumar and M. T. Coimbra, Heartbeat classification using morphological and dynamic features of ECG signals, *IEEE Trans. Biomed. Eng.* **59**(10) (2012) 2930–2941.
67. K. P. Murphy, *Machine Learning: A Probabilistic Perspective* (MIT press, 2012).
68. K. P. Bennett and C. Campbell, Support vector machines: Hype or hallelujah? *SIGKDD Explor. Newsl.* **2**(2) (2000) 1–13.
69. C. F. Borges and T. Pastva, Total least squares fitting of Bézier and B-spline curves to ordered data, *Comput. Aided Geom. Des.* **19**(4) (2002) 275–289.
70. A. Cornelio, E. L. Piccolomini and J. G. Nagy, Constrained numerical optimization methods for blind deconvolution, *Numer. Algorithms* **65**(1) (2014) 23–42.
71. D. Selimović, J. Lerga, P. Kovács and J. Prpić-Oršić, Improved parametrized multiple window spectrogram with application in ship navigation systems, *Digit. Signal Process.* **126** (2022) 103491–103504.
72. P. Kovács and A. M. Fekete, Nonlinear least-squares spline fitting with variable knots, *Appl. Math. Comput.* **354** (2019) 490–501.
73. K. M. R. Alam, N. Siddique and H. Adeli, A dynamic ensemble learning algorithm for neural networks, *Neural Comput. Appl.* **32** (2020) 8675–8690.