# Kernelising the Proportional Odds Model through Kernel Learning techniques

M. Pérez-Ortiz[a], P.A. Gutiérrez[a], M. Cruz-Ramírez[a], J. Sánchez-Monedero[a], C. Hervás-Martínez[a]

[a] *University of Córdoba, Dept. of Computer Science and Numerical Analysis*
*Rabanales Campus, Albert Einstein building, 14071 - Córdoba, Spain*

**Abstract**

The classification of patterns into naturally ordered labels is referred to as ordinal regression, which is a very common setting for real world applications. One of the most widely used ordinal regression algorithms is the Proportional Odds Model (POM), despite the linearity of the resultant decision boundaries. Through different proposals, this paper explores the notions of kernel trick and empirical feature space to reformulate the POM method and obtain nonlinear decision boundaries. Moreover, a new technique for aligning the kernel matrix taking into account the ordinal problem information is proposed, as well as a regularised gradient ascent methodology which is used to select the optimal dimensionality for the empirical feature space. The capability of the different developed methodologies is evaluated by the use of a nonlinearly separable toy dataset and an extensive set of experiments over 28 ordinal datasets. The results indicate that the tested methodologies are competitive with respect to other state-of-the-art algorithms, and they significantly improve the original POM algorithm.

*Keywords:*
Proportional Odds Model, Ordered logit, Ordinal Regression, Ordinal Classification, Kernel Trick, Kernel Learning

## 1. Introduction

In this paper, we consider the specific problem of ordinal regression, which shares properties of classification and regression settings. Formally, $\mathcal{Y}$ (the target space) is a finite set, but there exists an ordering among its elements. In contrast to regression, $\mathcal{Y}$ is a non-metric space, thus distances among categories are unknown. Besides, the zero-one loss function usually considered for standard classification does not reflect the ordering of $\mathcal{Y}$. Ordinal regression (or classification) problems arise in fields as information retrieval, preference learning, economy, and statistics, forming an emerging field in the areas of machine learning and pattern recognition.

A great number of statistical methods for categorical data treat all response variables as nominal, in such a way that the results are invariant to category permutations on those variables. However, there are many advantages in treating an ordered categorical variable as ordinal rather than nominal [1, 2]. In this vein, several approaches to tackle ordinal regression have been proposed in the domain of machine learning over the years, the Proportional Odds Model (POM) being one of the first ones, dating back to 1980 [3]. Indeed, the POM can be contextualised in the most popular framework for ordinal regression, *i.e.*, the threshold models [4, 5, 3], which are based on the assumption that an underlying real-valued outcome exists (also known as latent variable), although it is unobservable. These methods try to determine the nature of the underlying outcome by using a function $f(\cdot)$ and a set of thresholds to represent intervals in the range of $f(\cdot)$. Although very sophisticated and successful learning techniques have been recently developed for ordinal regression [6, 4, 5, 7], the use of the POM method is widespread. However, the resulting decision boundaries are linear, which is an unrealistic assumption for many real world problems. To deal with this issue, the proposals presented in this paper make use of the notion of the so-called kernel trick, which implicitly maps input patterns into a high-dimensional feature space via a function $\Phi(\cdot)$ in order to compute nonlinear decision boundaries. The standard process for applying the kernel trick requires reformulating the learning algorithm based on dot products between the different training points, which implies some difficulties in the case of the POM, as we will see. Alternatively, we consider the Empirical Feature Space (EFS) [8, 9], which preserves the geometrical structure of the original feature space (the dot products of the corresponding images are equal to the original kernel values, and the distances and angles in the feature space are uniquely determined by dot products). The EFS is Euclidean, this allowing the kernelisation of all kinds of linear machines [10, 11], with the advantage that the algorithm does not need to be formulated to deal with dot products.

The dimensionality of the EFS is the rank of the kernel matrix, which can be very high (*e.g.*, in the case of a Gaussian kernel it usually corresponds to the number of training patterns). This is a key factor in the reformulation of the POM algorithm, whose computational cost is closely related to the dimensionality of the dataset. Therefore, we propose different techniques to control this dimensionality while approximating the original

*This paper has been invited to be included in the "Special Issue Neurocomputing-IWANN2013".

information contained in the kernel matrix and therefore including some form of regularisation.

On the other hand, the performance of the POM model constructed in the EFS directly depends on how well the kernel function is adapted to the problem considered. Because of this, kernel-target alignment, a well-known kernel learning technique, [12, 13] is considered in this paper to better adapt the EFS to each dataset. This technique is extended by including ordinal weights in order to take the ordinal nature of the target variable into account. As will be analysed, such a kernel learning technique is very useful for creating a method to automatically compute the dimensionality of the EFS.

Summarising, the contributions of the paper can be said to be threefold: 1) the application of the EFS to compute nonlinear decision boundaries for the POM at a limited computational cost, leading naturally to probabilistic outputs; 2) an ordinal kernel learning technique to better match the different datasets; 3) an extension of this kernel learning technique in order to automatically decide the dimensionality of the EFS.

The kernelisation of the POM has also been considered in [14], where the POM method is extended for non-crisp ordinal regression task. Maximisation of the regularised loss function is accomplished by considering the representer theorem [15]. However, the paper makes reference to a different setting, where partial class memberships are given for the patterns, while we are provided with crisp ordinal targets. Moreover, our method approaches the optimisation of the model in a more direct way by redefining the model in the EFS. Other works have considered before a nonlinear version of the POM method (or more generally, a nonlinear version of logistic regression) by the use of artificial neural networks [16] or by including polynomial combinations of the input features. However, these strategies imply difficult optimisation processes. As will be shown in the experimental section, the use of the EFS with a Gaussian kernel allows the POM method to obtain much better results and to handle nonlinear decision boundaries. The experiments also show that the selection of the optimal dimensions is a crucial step which can significantly improve the algorithm performance, as well as the inclusion of the ordinal information in the kernel optimisation process.

The rest of the paper is organised as follows: Section 2 presents some useful previous notions; Section 3 shows a description of the different proposals; Section 4 describes the experimental study and analyses the results; and finally, Section 5 outlines some conclusions and future work.

## 2. Previous notions

The goal in classification is to assign an input vector $\mathbf{x}$ to one of $Q$ discrete classes $C_q, q \in \{1, \dots, Q\}$. A formal framework for the ordinal regression problem can be introduced considering an input space $\mathcal{X} \in \mathbb{R}^{m \times d}$, where $m$ is the number of training patterns and $d$ is the data dimensionality. Moreover, an outcome space $\mathcal{Y} = \{C_1, C_2, \dots, C_Q\}$ can be defined, where the labels are ordered in such a way that $C_1 \prec C_2 \prec \cdots \prec C_Q$, where $\prec$ denotes the order relation. The objective for this learning setting is to find a prediction rule $f : \mathcal{X} \to \mathcal{Y}$ by using an i.i.d. training sample $D = \{\mathbf{x}_i, y_i\}_{i=1}^m \in \mathcal{X} \times \mathcal{Y}$. The following subsections describe some of the concepts needed to understand the methodology proposed in this paper.

### 2.1. Proportional Odds Model

This is one of the first models specifically designed for ordinal regression, and it arises from a statistical background [3]. Let $h$ denote an arbitrary monotonic link function and $P(y \preceq C_q|\mathbf{x})$ the probability that a pattern $\mathbf{x}$ belongs to a class lower to $C_q$ (in the ordinal scale). The model:

$$h\Big(P(y \preceq C_q|\mathbf{x})\Big) = b_q - \boldsymbol{\beta}^\top \mathbf{x}, \quad q = 1, \dots, Q-1, \qquad (1)$$

links the cumulative probabilities to a linear predictor and imposes an stochastic ordering of the space $\mathcal{X}$, where $b_q$ is the threshold separating classes $C_q$ and $C_{q+1}$ and $\boldsymbol{\beta}$ is a linear projection. This model is naturally derived from the latent variable motivation; then instead of fitting a decision rule $f : \mathcal{X} \to \mathcal{Y}$ directly, this model defines a probability density function over the class labels for a given feature vector $\mathbf{x}$. Let us assume that the ordinal response comes from a coarsely measured latent continuous variable $f(\mathbf{x})$. Thus, label $C_q$ in the training set is observed if and only if $f(\mathbf{x}) \in [b_{q-1}, b_q]$, where the function $f$ (latent utility) and $b = \{b_0, b_1, ..., b_{Q-1}, b_Q\}$ are determined from data. By definition, $b_0 = -\infty$ and $b_Q = +\infty$ and the real line $f(\mathbf{x})$ is divided into $Q$ consecutive intervals, where each interval corresponds to a category $C_q$.

Now, let define a model of the latent variable, $f(\mathbf{x}) = \boldsymbol{\beta}^\top \mathbf{x} + \epsilon$, where $\epsilon$ is the random variable with zero expectation, $\mathbf{E}[\epsilon] = 0$, and distributed according to the distribution function $F_\epsilon$. Then, it follows that:

$$P(y \preceq C_q|\mathbf{x}) = \sum_{k=1}^q P(y = C_k|\mathbf{x}) = \sum_{k=1}^q P(f(\mathbf{x}) \in [b_{k-1}, b_k]) =$$

$$= P(f(\mathbf{x}) \in [-\infty, b_q]) = P(\boldsymbol{\beta}^\top \mathbf{x} + \epsilon \leq b_q) = P(\epsilon \leq b_q - \boldsymbol{\beta}^\top \mathbf{x}) =$$

$$= F_\epsilon(b_q - \boldsymbol{\beta}^\top \mathbf{x}).$$

If a distribution assumption $F_\epsilon$ is made for $\epsilon$, the cumulative model is obtained by choosing, as the inverse link function $h^{-1}$, the inverse distribution $F_\epsilon^{-1}$ (quantile function). Note that $F_\epsilon^{-1} : [0, 1] \to (-\infty, +\infty)$ is a monotonic function. The most common choice for $F_\epsilon$ is the logistic function [3].

### 2.2. Ideal kernel

Let $\mathcal{H}$ denote a high-dimensional or infinite-dimensional Hilbert space. Then, for any mapping of patterns $\Phi : \mathcal{X} \to \mathcal{H}$, the inner product $k(\mathbf{x}_i, \mathbf{x}_j) = \Big\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \Big\rangle_{\mathcal{H}}$ of the mapped inputs is known as a kernel function, giving rise to a positive semidefinite (PSD) matrix $\mathbf{K}$ for a given input set $\mathcal{X}$.

Although properties of a kernel function $k$ are important, often the kernel matrix ($\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$) plays a more important role than the kernel function, given that most kernel algorithms work with this matrix. Kernel matrices contain information about the similarity among the patterns in a dataset. Therefore, the empirical ideal kernel [13], $\mathbf{K}^*$, (*i.e.*, the matrix that

would represent perfect similarity information) will submit the following structure:

$$k^*(\mathbf{x}_i, \mathbf{x}_j) = \begin{cases} +1 & \text{if } y_i = y_j, \\ -1 & \text{otherwise} \end{cases} \quad (2)$$

where $\mathbf{K}^*_{ij} = k^*(\mathbf{x}_i, \mathbf{x}_j)$. Roughly speaking, $\mathbf{K}^*$ provides information about which patterns in the dataset should be considered as similar when performing some learning task. As we are dealing with a classification problem, patterns from the same class should be considered totally similar, while patterns from other classes should be considered as different as possible.

### 2.3. Centered kernel-target alignment

Suppose an ideal kernel matrix $\mathbf{K}^*$ and a given real kernel matrix $\mathbf{K}$. The underlying idea for kernel-target alignment (KTA) [13] is to choose the kernel matrix $\mathbf{K}$ (among a set of different matrices) closest to the ideal matrix $\mathbf{K}^*$. This can be evaluated by the Frobenius inner product between these matrices (*i.e.*, $\langle \mathbf{K}, \mathbf{K}^* \rangle_{\mathrm{F}} = \sum_{i,j=1}^{m} k(\mathbf{x}_i, \mathbf{x}_j) \cdot k^*(\mathbf{x}_i, \mathbf{x}_j)$), which give us information of how well the patterns are classified in his own category. Indeed, if we consider Eq. (2), the Frobenius inner product could be rewritten as $\langle \mathbf{K}, \mathbf{K}^* \rangle_{\mathrm{F}} = \sum_{y_i = y_j} k(\mathbf{x}_i, \mathbf{x}_j) - \sum_{y_i \neq y_j} k(\mathbf{x}_i, \mathbf{x}_j)$, where the term $\sum_{y_i = y_j} k(\mathbf{x}_i, \mathbf{x}_j)$ is related to the within-class distance, and the term $\sum_{y_i \neq y_j} k(\mathbf{x}_i, \mathbf{x}_j)$ is related to the between-class distance.

The KTA between two kernel matrices $\mathbf{K}$ and $\mathbf{K}^*$ is defined as:

$$\mathcal{A}(\mathbf{K}, \mathbf{K}^*) = \frac{\langle \mathbf{K}, \mathbf{K}^* \rangle_{\mathrm{F}}}{\sqrt{\langle \mathbf{K}^*, \mathbf{K}^* \rangle_{\mathrm{F}} \langle \mathbf{K}, \mathbf{K} \rangle_{\mathrm{F}}}}. \quad (3)$$

This quantity is totally maximised when the kernel function is capable to reflect the properties of the training dataset used to define the ideal kernel matrix.

However, some problems are found when considering KTA for datasets with skewed class distributions [13, 17]. These problems can be solved by the use of centred kernel matrices [12], leading a methodology (centred kernel-target alignment, CKTA) that have demonstrated to correlate better with performance than with the original definition of KTA. CKTA basically extends KTA by centring the patterns in the feature space. The centred kernel version of a matrix $\mathbf{K}$ can be written as:

$$\mathbf{K}_c = \mathbf{K} - \mathbf{K}\mathbf{1}_{\frac{1}{m}} - \mathbf{1}_{\frac{1}{m}}\mathbf{K} + \mathbf{1}_{\frac{1}{m}}\mathbf{K}\mathbf{1}_{\frac{1}{m}},$$

where $\mathbf{1}_{\frac{1}{m}}$ corresponds to a matrix with all the elements equal to $\frac{1}{m}$. $\mathbf{K}_c$ will also be a PSD matrix, fulfilling $k(\mathbf{x}, \mathbf{x}) \geq 0 \;\; \forall\, \mathbf{x} \in \mathcal{X}$ and symmetry.

### 2.4. Empirical Kernel Mapping

In this section, the Empirical Feature Space (EFS) [8] spanned by the training data is defined. By definition, a kernel matrix $\mathbf{K}$ can be diagonalised as follows:

$$\mathbf{K}_{(m \times m)} = \mathbf{P}_{(m \times r)} \cdot \mathbf{\Lambda}_{(r \times r)} \cdot \mathbf{P}^{\top}_{(r \times m)}, \quad (4)$$

where $r$ is the rank of $\mathbf{K}$, $\mathbf{\Lambda}$ is a diagonal matrix containing the $r$ non-zero eigenvalues of $\mathbf{K}$ in decreasing order (*i.e.*, $\lambda_1, \ldots, \lambda_r$), and $\mathbf{P}$ is a matrix consisting of the eigenvectors associated to those $r$ eigenvalues (*i.e.*, $\mathbf{u}_1, \ldots, \mathbf{u}_r$) in such a way that $\mathbf{K} = \sum_{i=1}^{r} \lambda_i \mathbf{u}_i \mathbf{u}_i^T$. Note that this mapping corresponds to the principal component analysis *whitening* step [18], but applied to the kernel matrix, instead of the covariance one. Then, the EFS can be defined as an Euclidean space preserving the dot product information about $\mathcal{H}$ contained in $\mathbf{K}$ (*i.e.*, this space is isomorphic to the embedded feature space $\mathcal{H}$, but being Euclidean). Since distances and angles of the vectors in the feature space are uniquely determined by dot products, the training data have the same geometrical structure in both the EFS and the feature space. The map from the input space to this $r$-dimensional EFS is defined as $\Phi_r^e : \mathcal{X} \to \mathbb{R}^r$. More specifically:

$$\Phi_r^e : \mathbf{x}_i \to \mathbf{\Lambda}^{-1/2} \cdot \mathbf{P}^{\top} \cdot (k(\mathbf{x}_i, \mathbf{x}_1), \ldots, k(\mathbf{x}_i, \mathbf{x}_m))^{\top}. \quad (5)$$

It can be checked that the kernel matrix of training images obtained by this transformation corresponds to $\mathbf{K}$, when considering the standard dot product [8, 9].

Furthermore, the EFS provides us with the opportunity to limit the dimensionality of the space by choosing the $j \leq r$ dominant eigenvalues (and their associated eigenvectors) to project the data, while maintaining the most important part of the structure of $\mathcal{H}$. Nevertheless, how to correctly choose $j$ is still a difficult issue to be solved.

Figure 1 has been included in order to graphically clarify the concept of EFS. It can be seen that, despite the fact that the three most representative dimensions are not enough to linearly separate the data, they actually provide useful information about the order of the classes and the separation between them.

## 3. Proposed methodology: Tackling the ordinal information via the kernel trick

Although Eq. (1) could be directly kernelised in the same vein that it is done with support vector machines (SVMs) (see, for example, [19]), this would imply substituting the standard hinge loss by the negative log likelihood loss (in our case, both adapted to ordinal regression). Because of the nature of this log likelihood loss function, this would reduce the sparsity of the obtained kernel machine. The reason then to consider the EFS is precisely to be able to reduce the dimensionality of the obtained kernel machine by the method presented in Section 3.2.1, which, in general, should improve the generalisation performance.

Three differentiated proposals can be found in this section of the paper. Firstly, we propose how to extend the POM method to deal with a nonlinear transformation of the input variables making use of the kernel trick (*i.e.*, the above mentioned EFS). Secondly, we reformulate the notion of CKTA (a common strategy for kernel learning) to deal with classification problems that present an ordinal structure by imposing different weights for the different similarity errors. Finally, a new method is proposed for reducing the dimensionality of the subspace to which the data are projected. As said before, this is very useful for the reformulated POM, because it can decrease a lot the computational complexity (as opposed to considering the EFS with the full-rank decomposition).

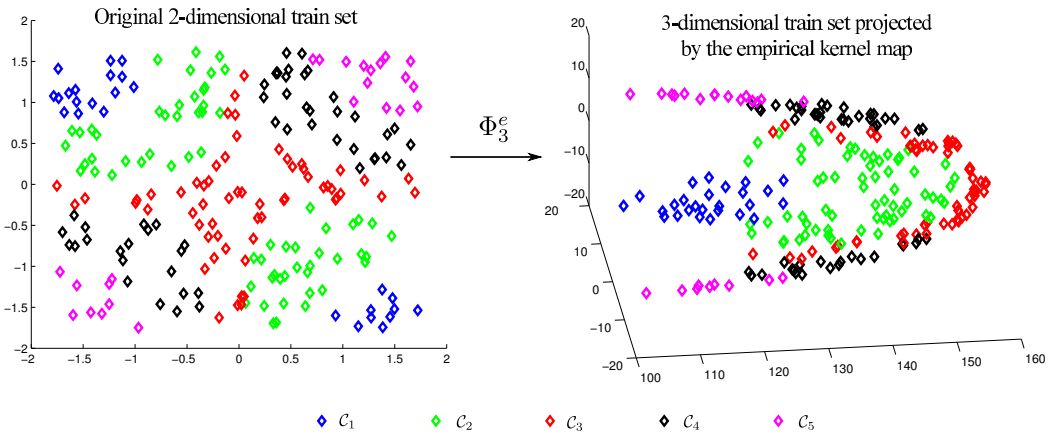


Figure 1: 3-dimensional approximation of the EFS induced by a Gaussian kernel for the nonlinearly separable synthetic toy dataset.

### 3.1. Proportional Odds Model in the Empirical Feature Space

Far beyond the definition of the EFS, it is well-known that the kernel trick turns a linear decision boundary in $\mathcal{H}$ into a nonlinear decision boundary in $\mathcal{X}$. This allows the formulation of nonlinear variants of many algorithms (those which can be cast in terms of the inner products between patterns). When using the EFS, this last restriction is avoided and any standard linear decision algorithm can be used, without any loss of generality. Figure 2 shows the case of a synthetic dataset representing a nonlinearly separable classification task and its transformation to the two-dimensional EFS (using the two most dominant eigenvectors), which is linearly separable.

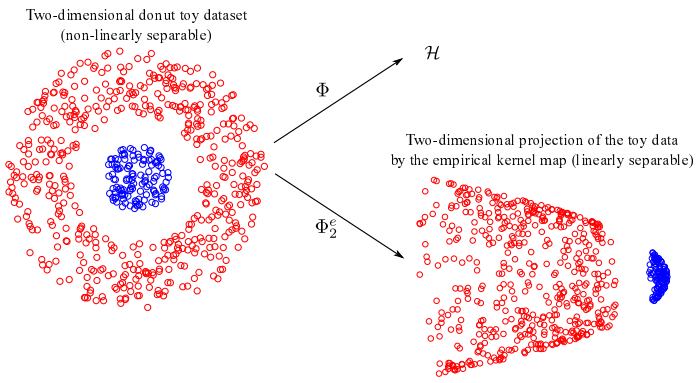


Figure 2: Synthetic two-dimensional dataset representing a nonlinearly separable classification problem and its transformation to the 2 dominant dimensions of the EFS induced by the Gaussian kernel function (linearly separable problem). Note that the $\mathcal{H}$ space can not be represented itself. However, the transformation performed when applying the kernel trick can be observed by analysing the two-dimensional EFS representation.

Now, consider the use of the EFS transformation $\Phi_r^e(\mathbf{x})$ (Eq. (5)) for redefining the POM. Eq. (1) is reformulated as:

$$h\left(P(y \preceq C_q|\mathbf{x})\right) = b_q - \boldsymbol{\beta}^\top \Phi_r^e(\mathbf{x}) = \tag{6}$$

$$= b_q - \boldsymbol{\beta}^\top \boldsymbol{\Lambda}^{-1/2} \cdot \mathbf{P}^\top \cdot (k(\mathbf{x}, \mathbf{x}_1), \ldots, k(\mathbf{x}, \mathbf{x}_m))^\top. \tag{7}$$

In this case, the model of the latent variable will submit the formulation $f(\Phi_r^e(\mathbf{x})) = \boldsymbol{\beta}^\top \cdot \Phi_r^e(\mathbf{x}) + \epsilon$, where $\boldsymbol{\beta}$ will be a linear projection. However, this projection will perform as a nonlinear decision function in $\mathcal{X}$, since a nonlinear transformation of the input variables is being used.

### 3.2. Kernel-Target Alignment for ordinal classification

Standard multinomial classification problems have been studied by using KTA based on a geometrical interpretation [20], resulting in a simple modification of the original KTA (which was initially designed for binary problems). Instead of considering the kernel equal to $-1$ when the patterns do not belong to the same class, it is assigned to $-1/(Q-1)$, being $Q$ the number of classes in the problem. This is done because each description $\mathbf{x}$ is associated to one of the $Q$ vertices of a $(Q-1)$-dimensional centred simplex. However, such approach is not consistent when considering a dataset with an ordinal structure, because all the errors committed are equally weighted and all the classes are said to be equally similar to the rest of classes.

For the sake of understanding, consider a dataset $D$ composed of five patterns belonging to four different classes, *i.e.*, $D = \{(\mathbf{x}_1, C_1), (\mathbf{x}_2, C_2), (\mathbf{x}_3, C_3), (\mathbf{x}_4, C_3), (\mathbf{x}_5, C_4)\}$. The ideal kernel matrix for $D$ can be seen in Table 1. Bold face is used in this Table to outline some of the entries of these matrices. Note that the kernel matrix can be seen from a pattern similarity/dissimilarity perspective. Now, examine the two arbitrary kernel matrices $\mathbf{K}_1$ and $\mathbf{K}_2$. In the Gram matrix $\mathbf{K}_1$, the pattern $\mathbf{x}_1 \in C_1$ is said to be similar to $\mathbf{x}_2 \in C_2$, while, in the Gram matrix $\mathbf{K}_2$, it is said to be similar to $\mathbf{x}_5 \in C_4$. In the case of ordinal regression, those misclassification errors involving a higher number of categories between the real label and the predicted one (in the ordinal scale) should be more penalised [2, 21, 22]. Similarly, matrix $\mathbf{K}_2$ (which is confusing a pattern from the first class with one of the fourth one) should result in a lower KTA than $\mathbf{K}_1$ (which is confusing this pattern with one of the neighbouring classes).

Table 2 shows three different error weighting cost matrices used in previous works. The first one is associated with the nominal classification setting, where all the misclassification errors are considered to be equal. The second one, is known as the absolute cost matrix, and it takes into account the difference of the assigned values for the categories, that is $|r(y_j) - r(y_i)|$, $(r(y_i)$ being the ranking for a given target $y_i$, *i.e.*, the position of $y_i$ in

Table 1: Example of different kernel matrices for the hypothetical dataset $D$.

| Ideal kernel matrix $\mathbf{K}^*$ | | | | | Kernel matrix $\mathbf{K}_1$ | | | | | Kernel matrix $\mathbf{K}_2$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| +1 | −1 | −1 | −1 | −1 | +1 | **+1** | −1 | −1 | −1 | +1 | −1 | −1 | −1 | **+1** |
| −1 | +1 | −1 | −1 | −1 | **+1** | +1 | −1 | −1 | −1 | −1 | +1 | −1 | −1 | −1 |
| −1 | −1 | +1 | +1 | −1 | −1 | −1 | +1 | +1 | −1 | −1 | −1 | +1 | +1 | −1 |
| −1 | −1 | +1 | +1 | −1 | −1 | −1 | +1 | +1 | −1 | −1 | −1 | +1 | +1 | −1 |
| −1 | −1 | −1 | −1 | +1 | −1 | −1 | −1 | −1 | +1 | **+1** | −1 | −1 | −1 | +1 |

the ordinal scale). Finally, the third one is the quadratic version of the absolute cost matrix. Absolute cost and quadratic absolute cost are commonly considered for ordinal regression problems, as a way of obtaining classifiers which minimise those misclassification errors involving several categories in the ordinal scale.

Table 2: Different cost matrices which can be found in the literature.

| | Nominal cost | | | | Absolute cost | | | | Quadratic abs. cost | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $C_1$ | $C_2$ | $C_3$ | $C_4$ | $C_1$ | $C_2$ | $C_3$ | $C_4$ | $C_1$ | $C_2$ | $C_3$ | $C_4$ |
| $C_1$ | 0 | 1 | 1 | 1 | 0 | 1 | 2 | 3 | 0 | 1 | 4 | 9 |
| $C_2$ | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 2 | 1 | 0 | 1 | 4 |
| $C_3$ | 1 | 1 | 0 | 1 | 2 | 1 | 0 | 1 | 4 | 1 | 0 | 1 |
| $C_4$ | 1 | 1 | 1 | 0 | 3 | 2 | 1 | 0 | 9 | 4 | 1 | 0 |

In the same vein, we propose to consider these matrices when obtaining the KTA of a matrix, in order to penalise differently the misalignment errors of an evaluated matrix. That is, a weighting matrix $\mathbf{W}$ is defined in such a way that $\mathbf{K}^* \circ \mathbf{W}$ imposes a weighting for the different similarity or dissimilarity errors committed, where $\mathbf{A} \circ \mathbf{B}$ represents the hadamard or entrywise product between matrices $\mathbf{A}$ and $\mathbf{B}$. A first idea for weighting errors would be the use of the absolute errors commonly used for ordinal classification, *i.e.*:

$$w(\mathbf{x}_i, \mathbf{x}_j) = \begin{cases} 1, & \text{if } y_i = y_j, \\ |r(y_i) - r(y_j)|, & \text{otherwise.} \end{cases} \tag{8}$$

As discussed previously, the centred version of the matrices will be considered, avoiding problems with skewed class distributions. Therefore, the proposed ordinal version $\tilde{\mathcal{A}}_c$ of CKTA is defined as follows:

$$\tilde{\mathcal{A}}_c(\mathbf{K}, \mathbf{K}^*) = \mathcal{A}_c(\mathbf{K}, \mathbf{K}^* \circ \mathbf{W}). \tag{9}$$

This reformulation of CKTA for ordinal problems can be used for optimising the parameters of the kernel matrix (subsection 3.2.1), as well as for choosing the optimal dimensions for projecting the data onto a lower dimensional space (subsection 3.3). Both approaches will be considered for the experiments in order to improve the quality of the EFS in conjunction with the POM method.

#### 3.2.1. *Optimisation of the ordinal Centred Kernel-Target Alignment via Multiple Kernel Learning*

For the optimisation of the proposed ordinal CKTA, one could use any of the optimisation strategies proposed for the original CKTA. In this paper, we will use two different strategies. This subsection presents one of them, and subsection 3.3 proposes the other. In this subsection, we use a Quadratic Programming problem (QP) (by means of multiple kernel learning techniques), which has a single global maximum and it is easier to optimise. The solution of this QP problem will result in a kernel matrix defining the optimal EFS for the considered problem. We optimise a convex combination of kernel matrices, where each matrix is associated to a different parameter for the kernel width. Therefore, we fix a set of $p$ possible parameter values for the kernel width $\alpha$, *i.e.*, $\{\alpha_1, \ldots, \alpha_p\}$ and compute the kernel matrices obtained for these values $\{\mathbf{K}_{\alpha_1}, \ldots, \mathbf{K}_{\alpha_p}\}$. To optimise CKTA (or the proposed ordinal version), we can derive a kernel matrix $\mathbf{K}_{\boldsymbol{\delta}} = \sum_{i=1}^{p} \delta_i \mathbf{K}_{\alpha_i}$ with $\delta_i \geq 0$ and $\sum_{i=1}^{p} \delta_i = 1$. The optimisation problem for the ordinal version of CKTA will be the following:

$$\max_{\boldsymbol{\delta} \in \mathcal{M}} \frac{\langle \mathbf{K}_{\boldsymbol{\delta}_c}, \mathbf{K}^* \circ \mathbf{W} \rangle_{\mathrm{F}}}{\|\mathbf{K}_{\boldsymbol{\delta}_c}\|_{\mathrm{F}}},$$

where $\|\mathbf{A}\|_{\mathrm{F}} = \sqrt{\langle \mathbf{A}, \mathbf{A} \rangle_{\mathrm{F}}}$ and $\mathcal{M} = \{\boldsymbol{\delta} : \|\boldsymbol{\delta}\|_2 = 1\}$. The QP optimization problem associated can be solved as in [12].

To show how the different weights in Table 1 may influence the choice of the parameters we include the optimisation surfaces obtained for $\boldsymbol{\delta}$ when $\boldsymbol{\alpha} = \{0.01, 1, 100\}$. We use a three-dimensional simplex (to fulfil $\delta_i \geq 0$ and $\sum_{i=1}^{3} \delta_i = 1$), as can be seen in Figure 3, where the coloured points show how to select the values of the parameters $\boldsymbol{\delta}$ in order to fulfil their constraints. Figure 4 shows these optimisation surfaces for two datasets of the experiments considered in this paper (LEV and toy) and the three weight matrices in Table 1. As can be appreciated, the surfaces are very different and the optimum value can be found in a different region of the simplex.

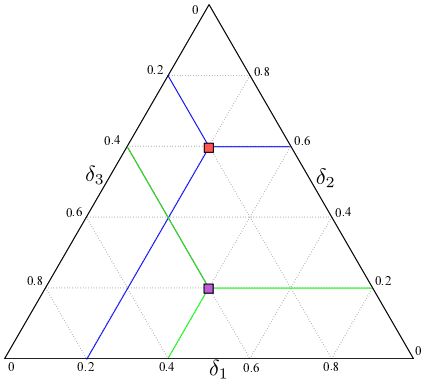


Figure 3: Simplex example where it can be seen how to compute $\delta_1$, $\delta_2$ and $\delta_3$ in order to fulfill the constraints $\delta_i \geq 0$ and $\sum_{i=1}^{3} \delta_i = 1$.

### 3.3. *Selection of bases for projecting: A regularised gradient-based technique using CKTA*

The above mentioned methodology is not suitable for choosing the optimal set of bases for projecting the data. Therefore, once the kernel matrix has been optimized by the process presented in the previous subsection, we now present a regularised gradient ascent methodology to improve the alignment of the kernel matrix by selecting some of its bases.

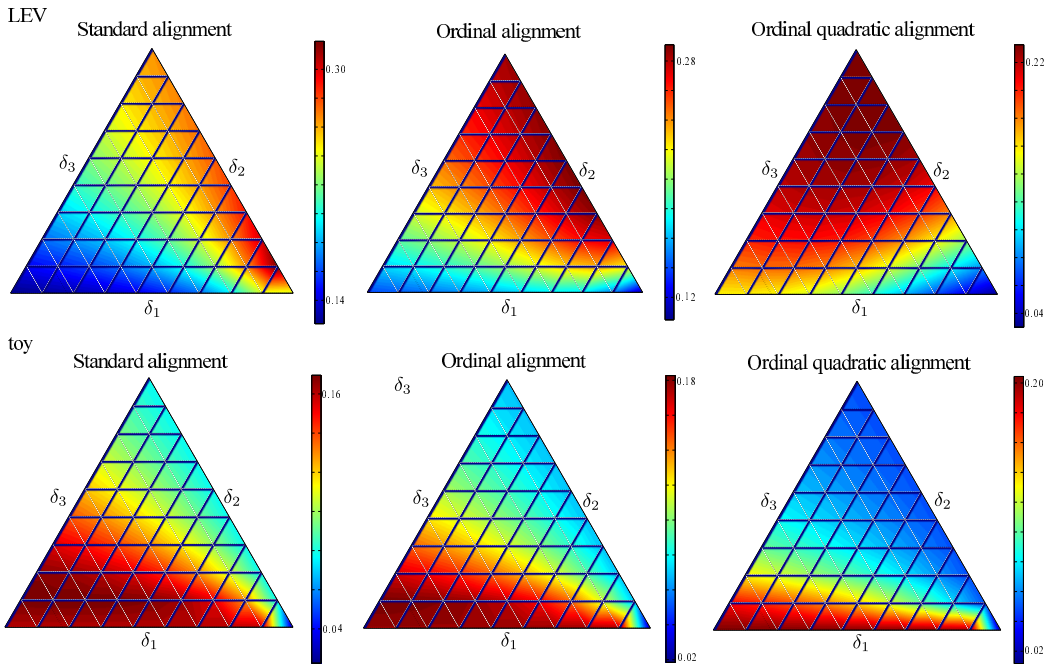


Figure 4: Kernel-target alignment optimisation surfaces for $\{\delta_1, \delta_2, \delta_3\}$ for the LEV and toy datasets and three different weighting matrices (see Table 2).

Usually, the $j$ dominant eigenvectors (the ones associated to the highest eigenvalues) are used as a projection onto a subspace to remove noise (as done in principal components analysis) or for visualisation purposes. Therefore, the eigenvalues ranking from $j+1$ to $r$ (and the corresponding eigenvectors) are discarded so that $\mathbf{K}_j = \sum_{i=1}^{j} \lambda_i \mathbf{u}_i \mathbf{u}_i^{\mathrm{T}}$. By using this idea, the distance to the original kernel $r$-rank matrix $\mathbf{K}$ (*i.e.*, $\|\mathbf{K} - \mathbf{K}_j\|_{\mathrm{F}}^2$) is minimised over all rank-$j$ matrices. However, in this case, we aim to find the projection that minimises $\|\mathbf{K}^* - \mathbf{K}_j\|_{\mathrm{F}}^2$ for $\mathbf{K}^*$ being the ideal kernel. Note that, since $\mathbf{K}$ does not include any information about the target variable, the bases associated to the highest eigenvalues of $\mathbf{K}$ do not have to be so informative. Alternatively, we can form a matrix:

$$\mathbf{K}_{\mathbf{w}} = \sum_{i=1}^{r} f(w_i) \lambda_i \mathbf{u}_i \mathbf{u}_i^T, \tag{10}$$

where $f(w_i) \in [0, 1]$ so as to maintain $\mathbf{K}_{\mathbf{w}}$ to be PSD and for simplicity. In this way, the weight of the eigenvectors is now determined by $f(w_i)$ and $\lambda_i$. The objective of this definition is to generalise the combination of the eigenvectors in order to obtain information about which of them are more important for improving the CKTA. We aim to find a lower subspace for our data that maintains the labelling information in a proper way. We propose to define the optimisation problem as follows:

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \left( \mathcal{A}_c(\mathbf{K}_{\mathbf{w}}, \mathbf{K}^*) - \frac{\mu}{r} \sum_{i=1}^{r} f(w_i) \right), \tag{11}$$

where $f(w_i) \in [0, 1]$ and $\mu$ is a regularisation parameter. $L^1$ or $L^2$ norms could be considered for the weights (*i.e.*, $f(w_i) = |w_i|$ or $f(w_i) = w_i^2$, respectively). For simplicity, we choose:

$$f(w_i) = \frac{1}{1 + e^{-w_i}}, \tag{12}$$

*i.e.*, the sigmoid function. We experimentally found that this formulation promotes more sparsity than $L^2$ norm, while being still derivable. As we apply gradient descent optimisation for optimisation, considering $L^1$-norm would imply a constrained problem (with a higher computational cost) or applying iterative techniques similar to those in [23].

Because of the differentiability of the function to maximise in Eq. (11) (which will be named $g$ from now on) with respect to the vector $\mathbf{w}$, a gradient ascent algorithm can be used to maximise it. The gradient vector will be composed of the following partial derivatives $\nabla g = \left[\frac{\partial g}{\partial w_1}, \ldots, \frac{\partial g}{\partial w_r}\right]^{\mathrm{T}}$. The iRprop$^+$ algorithm is considered for optimising the aforementioned function, because of its proven robustness for optimising KTA [24]. Each parameter $w_i$ will be updated considering the sign of $\frac{\partial g}{\partial w_i}$ but not the magnitude. Although the second partial derivatives can also be computed and used for optimisation, they actually make the process more computationally costly due to the complexity of their formulae.

The first derivative of $g$ with respect to $w_i$ is:

$$\frac{\partial g}{\partial w_i} = \frac{\partial \mathcal{A}_{\mathrm{c}}(\mathbf{K}_{\mathbf{w}}, \mathbf{K}^*)}{\partial w_i} - \frac{\mu}{r} \cdot \frac{\partial f}{\partial w_i}, \tag{13}$$

where the alignment derivative with respect to $w_i$ is:

$$\frac{\partial \mathcal{A}_{\mathrm{c}}(\mathbf{K}_{\mathbf{w}}, \mathbf{K}^*)}{\partial w_i} = \tag{14}$$

$$= \frac{1}{\|\mathbf{K}_{\mathrm{c}}^*\|_{\mathrm{F}}} \left[ \frac{\left\langle \frac{\partial \mathbf{K}_{\mathbf{w}}}{\partial w_i}, \mathbf{K}_{\mathrm{c}}^* \right\rangle_{\mathrm{F}}}{\|\mathbf{K}_{\mathbf{w}_{\mathrm{c}}}\|_{\mathrm{F}}} - \frac{\langle \mathbf{K}_{\mathbf{w}}, \mathbf{K}_{\mathrm{c}}^* \rangle_{\mathrm{F}} \cdot \left\langle \mathbf{K}_{\mathbf{w}_{\mathrm{c}}}, \frac{\partial \mathbf{K}_{\mathbf{w}}}{\partial w_i} \right\rangle_{\mathrm{F}}}{\|\mathbf{K}_{\mathbf{w}_{\mathrm{c}}}\|_{\mathrm{F}}^3} \right], \tag{15}$$

and, for matrices $\mathbf{K}_1$ and $\mathbf{K}_2$, it is satisfied that $\langle \mathbf{K}_{1_{\mathrm{c}}}, \mathbf{K}_{2_{\mathrm{c}}} \rangle_{\mathrm{F}} = \langle \mathbf{K}_1, \mathbf{K}_{2_{\mathrm{c}}} \rangle_{\mathrm{F}} = \langle \mathbf{K}_{1_{\mathrm{c}}}, \mathbf{K}_2 \rangle_{\mathrm{F}}$ [12], which simplifies the computation. The computation of the KTA takes $O(m^2)$ operations per

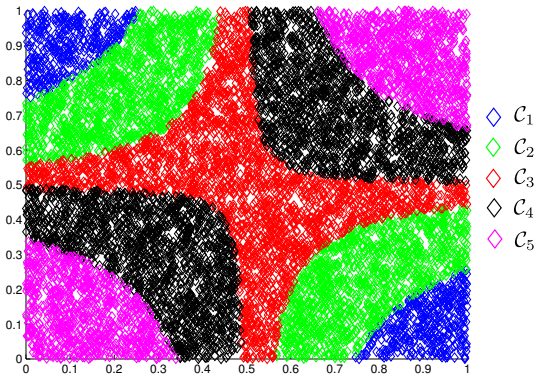


Figure 5: Two-dimensional representation of the structure of an ordinal nonlinearly separable toy dataset.

parameter $w_i$ to optimise [25]. Because this optimisation does not involve any additional optimisation problem, it is very fast in practice. The derivative of the kernel (see Eq. (10)) is in this case:

$$\frac{\partial \mathbf{K_w}}{\partial w_i} = \frac{\partial f}{\partial w_i}\lambda_i \mathbf{u}_i \mathbf{u}_i^T = \frac{e^{-w_i}}{(1+e^{-w_i})^2}\lambda_i \mathbf{u}_i \mathbf{u}_i^T. \quad (16)$$

The regularisation term in Eq. (11) (*i.e.*, the term $\mu \sum_{i=1}^{r} f(w_i)$) results in the less representative bases presenting a $w_i$ value close to zero, and the most representative ones close to one. The bases with $w_i$ close to one are the ones chosen for projecting the data. The parameter $\mu$ is set by cross-validation. After applying the gradient ascent method, those bases which $f(w_i) < 10^{-6}$ are eliminated from the kernel matrix, and the original eigenvalues of the remaining bases are taken into account to reconstruct the matrix.

Although we have considered original CKTA for all these definitions, ordinal CKTA can be similarly considered by applying the weight matrix to the ideal kernel matrix. If we include the ordinal version of CKTA in Eq. (11), the projections that better maintain the ordinal similarity information will be preferred.

The convergence of the proposal to a proper solution is now discussed. Consider the case when there is a basis that perfectly projects the patterns according to the labelling (for example, the eigenvalue $\lambda_1$ and the associated eigenvector $\mathbf{u}_1$). Let denote the projected kernel matrix using this basis as $\mathbf{K}_{\lambda_1} = f(w_1)\lambda_1 \mathbf{u}_1 \mathbf{u}_1^T$. Furthermore, consider other basis ($\lambda_2$, $\mathbf{u}_2$ and $\mathbf{K}_{\lambda_2}$) which projection results in no useful information about the labelling. That is, $\mathcal{A}_c(\mathbf{K}_{\lambda_1}, \mathbf{K}^*) = 1$ (or, in other words $\mathbf{K}_{\lambda_1} = c\mathbf{K}^*$, where $c$ is a scalar) and $\mathcal{A}_c(\mathbf{K}_{\lambda_2}, \mathbf{K}^*) = 0$. Let suppose $r = 2$, so that $\mathbf{K_w} = \mathbf{K}_{\lambda_1} + \mathbf{K}_{\lambda_2}$. Under these assumptions, what we would like to show is how $f(w_2)$ influences $\mathcal{A}_c(\mathbf{K_w}, \mathbf{K}^*)$. First, the alignment can also be defined as:

$$\mathcal{A}_c(\mathbf{K_w}, \mathbf{K}^*) = \frac{\mathrm{Tr}(\mathbf{K_w}\mathbf{K}_c^*)}{\sqrt{\mathrm{Tr}((\mathbf{K}_c^*)^2)\mathrm{Tr}(\mathbf{K}_{\mathbf{wc}}^2)}} = \frac{\mathrm{Tr}(\mathbf{K}_{\lambda_1}\mathbf{K}_c^*) + \mathrm{Tr}(\mathbf{K}_{\lambda_2}\mathbf{K}_c^*)}{\sqrt{\mathrm{Tr}((\mathbf{K}_c^*)^2)\mathrm{Tr}(\mathbf{K}_{\mathbf{wc}}^2)}}. \quad (17)$$

Note that the fact that $\mathcal{A}_c(\mathbf{K}_{\lambda_2}, \mathbf{K}^*) = 0$ comes from $\mathrm{Tr}(\mathbf{K}_{\lambda_2}\mathbf{K}_c^*) = 0$. Besides, $\mathrm{Tr}(\mathbf{K}_{\mathbf{wc}}^2) = (f(w_1)\lambda_1)^2 + (f(w_2)\lambda_2)^2$. Note that the only case in which $\mathcal{A}_c(\mathbf{K_w}, \mathbf{K}^*) = \mathcal{A}_c(\mathbf{K}_{\lambda_1}, \mathbf{K}^*)$ (*i.e.*, the maximum value) is for $f(w_2) = 0$, which makes $\mathrm{Tr}(\mathbf{K_w}\mathbf{K}_c^*) = \mathrm{Tr}(\mathbf{K}_{\mathbf{wc}}^2)$.

Moreover, given the definition of $\mathbf{K}_{\lambda_1}$ and $\mathbf{K}_{\lambda_2}$ and the fact that $\mathbf{u}_1$ and $\mathbf{u}_2$ are orthonormal, *i.e.*, $\mathrm{Tr}(\mathbf{K}_{\lambda_1 c}\mathbf{K}_{\lambda_2 c}) = 0$, the alignment between these two matrices is zero (they are uncorrelated). In this way, the alignment provided by one basis does not affect the alignment provided by the others.

Finally, note that this projection technique can also be used for visualisation purposes in supervised learning contexts (as an analogue technique for Kernel Principal Component Analysis for non-supervised problems), by optimising the bases and then representing the projection onto the two or three most dominant bases.

Table 3: Characteristics of the 28 benchmark datasets used for the experiments.

| Dataset | #Pat. | #Attr. | #Classes | Class distribution |
|---|---|---|---|---|
| contact-lenses | 24 | 6 | 3 | (15,5,4) |
| pasture | 36 | 25 | 3 | (12,12,12) |
| squash-stored | 52 | 51 | 3 | (23,21,8) |
| squash-unstored | 52 | 52 | 3 | (24,24,4) |
| tae | 151 | 54 | 3 | (49, 50, 52) |
| SWD | 1000 | 10 | 4 | (32,352,399,217) |
| car | 1728 | 21 | 4 | (1210,384,69,65) |
| diabetes5 | 43 | 2 | 5 | (5,6,22,8,2) |
| pyrim5 | 74 | 27 | 5 | (7,28,17,12,10) |
| triazines5 | 186 | 60 | 5 | (7,10,26,86,57) |
| wisconsin5 | 194 | 32 | 5 | (67,41,43,24,19) |
| machine5 | 209 | 6 | 5 | (152,27,13,7,10) |
| toy | 300 | 2 | 5 | (35,87,79,68,31) |
| auto5 | 392 | 7 | 5 | (91,131,101,59,10) |
| housing5 | 506 | 13 | 5 | (77,239,123,36,31) |
| eucalyptus | 736 | 91 | 5 | (180, 107, 130, 214, 105) |
| stock5 | 950 | 9 | 5 | (158,227,272,207,86) |
| LEV | 1000 | 4 | 5 | (93,280,403,197,27) |
| automobile | 205 | 71 | 6 | (3,22,67,54,32,27) |
| heating | 768 | 8 | 8 | (20,265,112,51,119,85,82,34) |
| cooling | 768 | 8 | 8 | (150,198,52,114,126,89,26,13) |
| diabetes10 | 43 | 2 | 10 | (2,3,3,3,10,12,4,2,2,2) |
| pyrim10 | 74 | 27 | 10 | (2,2,14,14,13,5,10,4,3,7) |
| triazines10 | 186 | 60 | 10 | (4,3,2,8,11,15,36,50,45,12) |
| wisconsin10 | 194 | 32 | 10 | (46,21,28,13,25,18,14,10,9,10) |
| machine10 | 209 | 6 | 10 | (115,37,21,6,8,5,3,4,4,6) |
| auto10 | 392 | 7 | 10 | (13,78,73,58,53,48,37,22,4,6) |
| housing10 | 506 | 13 | 10 | (22,55,85,154,84,39,29,7,10,21) |
| stock10 | 950 | 9 | 10 | (48,110,108,119,168,104,104,103,64,22) |

All nominal variables are transformed into binary ones.
For discretised datasets, the number included in their names (5 or 10) represents the number of bins considered during discretisation.

## 4. Experimental results

This section presents the experimental part of the paper: the datasets and methods tested, the evaluation measures and, finally, the results obtained.

### *4.1. Datasets*

Several benchmark datasets have been considered in order to validate the methodologies proposed; some publicly available real ordinal classification datasets were extracted from the UCI and `mldata.org` repositories [26, 27] and some of the ordinal regression benchmark datasets provided by Chu et. al [28] were considered due to their widespread use in ordinal regression [5, 29]. The latter do not originally represent ordinal

classification tasks but regression ones, where the target variable is discretised into $Q$ different bins (representing classes) with equal bining, to turn regression into ordinal classification. Table 3 presents the main characteristics of 28 the datasets used for the experiments. A synthetic two-dimensional toy dataset has been included in the experiments. The representation of this dataset can be seen in Figure 5.

Regarding the experimental setup, a 30-holdout stratified technique has been applied to divide the datasets, using 75% of the patterns for training the model, and the remaining 25% for testing it. One model is obtained and evaluated for each split. Finally, the results are taken as the mean and standard deviation over each one of the test sets.

### *4.2. Metrics considered*

Concerning evaluation measures, several metrics can be considered for ordinal classifiers, but the most common ones in machine learning are the Mean Absolute Error ($MAE$) and the Accuracy ($Acc$) [2, 5, 29], where the $MAE$ is the average deviation in absolute value of the predicted class from the true class [21], $MAE = (1/N)\sum_{i=1}^{N} e(\mathbf{x}_i)$, where $e(x_i) = |r(y_i) - r(y_i^*)|$ is the distance between the true and the predicted ranks. $MAE$ values range from 0 to $Q-1$ (maximum deviation in number of ranks between two labels). Instead, $Acc$ penalises all mistakes equally.

### *4.3. Methods tested*

To test the different proposals in Section 3, we consider the comparison of the following methods:

- The POM algorithm in the original input space $\mathcal{X}$, which is a linear method (POM).
- A kernelisation of the POM algorithm, cross-validating the number of dimensions for the projected subspace and the width of the Gaussian kernel (K-POM). The dimensions selected are always those with the highest eigenvalues. For this and the following three methods, the EFS was considered to perform this kernelisation, as introduced in Section 3.1.
- The POM algorithm kernelised using a regularised gradient-based technique for selecting the dominant dimensions (KRGB-POM). The width of the Gaussian kernel is also selected through cross-validation, so the difference between KRGB-POM and K-POM lies only on the selection of the dominant dimensions through the regularised gradient-based technique presented in Section 3.3.
- Kernelised version of the POM algorithm solving a QP optimisation problem for learning the kernel presented in Section 3.2.1 (instead of cross-validation), and the regularised gradient-based technique for selecting the dominant dimensions of Section 3.3 (KLRGB-POM). The original version of CKTA was considered.
- Finally, we also tested the KLRGB-POM methodology described above, but considering the notion of ordinal CKTA (introduced in Section 3.2) for both kernel optimisations (OKLRGB-POM).

The source code in Matlab for the proposed methods can be downloaded from the web associated to this paper[1].

Furthermore, two well-known kernel methods for ordinal regression have been chosen for comparison purposes (Kernel Discriminant Learning for Ordinal Regression [5], KDLOR, and Support Vector for Ordinal Regression with Implicit Constraints [29], SVORIM).

For model selection, a stratified nested 3-fold cross-validation has been applied to the training sets, with kernel width within the values $\{10^{-2}, 10^{0}, 10^{2}\}$. The same values are considered for the cost parameter of SVORIM. The cross-validation criterion is the $MAE$, since it can be considered the most common one in ordinal regression. The kernel selected for all the algorithms is the Gaussian one, $K(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{\sigma^2}\right)$ where $\sigma$ is the width of the kernel. The logit function has been used for all the POM-based algorithms. The number of dimensions for the empirical feature space ($j$) has been cross-validated within the values $\{10, 20, 30\}$. For KLRGB-POM and OKLRGB-POM, the number of kernels and widths considered ($p$) are the same than those used for cross-validation ($\{10^{-2}, 10^{0}, 10^{2}\}$).

Concerning the gradient-based technique, the initial points for all the methods tested were randomly chosen from a uniform distribution $U[0, 1]$. The gradient norm stopping criterion was set at $10^{-5}$ and the maximum number of conjugate gradient steps at 50. Furthermore, the $\mu$ parameter associated to the regularisation was cross-validated within the values $\{10^{-4}, 10^{-2}, 10^{0}\}$.

### *4.4. Results*

The results of the battery of experiments can be seen in Table 4 (for $Acc$) and Table 5 (for $MAE$), where all the methods described in the previous subsection have been tested. To better summarise these results, these tables also show the test mean rankings in terms of $Acc$ and $MAE$ for all the methods considered in this experiment, along all of the 28 datasets. For each dataset, a ranking of 1 is given to the best method in average, and a 7 is given to the worst one. It can be seen, that simply cross-validating the number of dimensions, the POM algorithm can be improved to a great extent (POM versus K-POM comparison), and that the use of a more intelligent technique for selecting the bases for projecting is a good option (KRGB-POM versus K-POM). The QP kernel learning technique of methods KLRGB-POM and OKLRGB-POM also improve the results. Finally, it can be seen that using a weighting matrix in CKTA can help to improve the results in terms of $MAE$ (KLRGB-POM versus OKLRGB-POM). The results also show that the proposals are competitive with the selected ordinal state-of-the-art methods (KDLOR and SVORIM) and are able to outperform the standard linear POM algorithm in most cases. The cases of the toy and eucalyptus datasets are very good examples of the

[1] `http://www.uco.es/grupos/ayrna/neucom-kpom`

Table 4: Results obtained for each method reported in terms of *Acc*.

| Dataset | POM | K-POM | KRGB-POM | KLRGB-POM | OKLRGB-POM | KDLOR | SVORIM |
|---|---|---|---|---|---|---|---|
| contact-lenses | **65.56 ± 15.74** | 50.00 ± 20.53 | 60.00 ± 18.88 | 62.23 ± 13.79 | 62.22 ± 13.79 | 48.89 ± 21.41 | *63.89 ± 8.84* |
| pasture | 46.30 ± 13.40 | 60.74 ± 16.18 | 63.33 ± 13.42 | 64.44 ± 13.18 | *65.19 ± 11.57* | 60.74 ± 11.20 | **66.67 ± 11.30** |
| squash-stored | 38.97 ± 15.38 | 60.51 ± 12.73 | **69.49 ± 12.20** | *66.15 ± 11.36* | 65.13 ± 11.02 | 65.38 ± 13.36 | 63.33 ± 11.37 |
| squash-unstored | 36.67 ± 14.66 | 62.05 ± 14.27 | 76.15 ± 11.13 | **77.95 ± 12.08** | *76.92 ± 12.29* | 73.33 ± 13.81 | 75.38 ± 12.02 |
| tae | 43.86 ± 11.49 | 36.32 ± 6.00 | 57.28 ± 7.17 | **57.89 ± 6.37** | *57.46 ± 6.26* | 57.28 ± 5.43 | 57.19 ± 6.87 |
| SWD | 52.88 ± 3.22 | 55.81 ± 2.84 | *57.28 ± 3.38* | **57.51 ± 3.45** | 57.09 ± 3.61 | 47.93 ± 2.98 | 56.73 ± 2.78 |
| diabetes5 | 42.73 ± 10.98 | 41.52 ± 14.46 | 44.55 ± 13.58 | 48.18 ± 8.32 | *49.39 ± 7.80* | **52.42 ± 5.69** | 49.09 ± 7.40 |
| pyrim5 | 46.67 ± 10.95 | *59.90 ± 8.13* | 57.37 ± 9.23 | 57.72 ± 8.67 | **60.00 ± 9.11** | 49.12 ± 10.82 | 58.77 ± 9.88 |
| triazines5 | 30.85 ± 1.08 | 45.39 ± 3.84 | 42.70 ± 10.04 | 44.40 ± 5.05 | 44.82 ± 4.36 | **46.60 ± 2.46** | *45.96 ± 2.66* |
| wisconsin5 | *28.50 ± 6.33* | **29.52 ± 4.24** | 24.76 ± 8.11 | 20.95 ± 3.51 | 20.75 ± 3.64 | 21.22 ± 1.48 | 26.33 ± 4.89 |
| machine5 | **85.16 ± 4.28** | 83.84 ± 5.31 | 83.33 ± 5.03 | 82.83 ± 5.50 | *83.90 ± 4.02* | 82.14 ± 4.40 | 83.58 ± 3.87 |
| toy | 30.13 ± 5.36 | 91.15 ± 3.44 | *92.09 ± 3.05* | 91.16 ± 3.14 | 90.84 ± 3.60 | 88.93 ± 3.17 | **94.76 ± 2.57** |
| auto5 | 62.11 ± 5.04 | **75.82 ± 3.77** | *75.37 ± 4.15* | 74.32 ± 4.60 | 74.08 ± 4.92 | 70.99 ± 4.55 | 74.76 ± 3.37 |
| housing5 | 60.68 ± 4.09 | 73.12 ± 4.08 | 73.81 ± 4.05 | *76.17 ± 2.88* | **76.98 ± 2.80** | 73.81 ± 3.14 | 75.51 ± 3.17 |
| eucalyptus | 15.02 ± 1.51 | 52.92 ± 3.47 | 55.60 ± 3.17 | **61.54 ± 2.43** | *61.07 ± 5.56* | 56.90 ± 3.69 | 60.69 ± 2.58 |
| stock5 | 63.77 ± 2.25 | 84.24 ± 1.86 | 87.59 ± 1.67 | *88.98 ± 1.68* | **89.02 ± 1.98** | 85.28 ± 2.05 | 87.87 ± 1.83 |
| LEV | 53.92 ± 3.18 | 61.95 ± 2.69 | 62.15 ± 2.41 | *62.40 ± 2.77* | **62.60 ± 2.81** | 54.60 ± 2.88 | 61.72 ± 2.87 |
| automobile | 38.78 ± 20.14 | 53.59 ± 5.82 | 69.62 ± 6.69 | 64.87 ± 4.89 | 65.90 ± 5.39 | **70.71 ± 6.32** | *70.71 ± 4.19* |
| heating | 53.02 ± 3.80 | 69.41 ± 2.70 | **86.77 ± 2.89** | 81.25 ± 3.78 | *82.41 ± 1.91* | 78.23 ± 6.64 | 74.74 ± 5.43 |
| cooling | 50.36 ± 3.84 | 64.84 ± 2.41 | **73.63 ± 3.01** | 71.04 ± 2.49 | 70.82 ± 2.65 | *72.64 ± 3.49* | 68.61 ± 5.02 |
| diabetes10 | **25.15 ± 11.62** | 20.30 ± 9.75 | *23.03 ± 8.85* | 22.73 ± 7.83 | *23.03 ± 8.85* | 19.09 ± 8.04 | 20.00 ± 10.78 |
| pyrim10 | 32.81 ± 10.41 | *34.21 ± 10.95* | 30.18 ± 11.80 | 22.64 ± 5.01 | 22.63 ± 5.55 | 30.18 ± 8.73 | **37.72 ± 7.58** |
| triazines10 | 6.23 ± 0.80 | 29.29 ± 4.73 | 24.18 ± 10.42 | **30.35 ± 6.34** | 29.15 ± 7.56 | 23.76 ± 4.26 | *29.86 ± 3.28* |
| wisconsin10 | **15.92 ± 5.48** | *12.65 ± 4.37* | 12.45 ± 5.98 | 10.82 ± 2.99 | 10.61 ± 3.69 | 6.53 ± 0.83 | 11.22 ± 3.93 |
| machine10 | 64.84 ± 6.51 | *67.99 ± 4.87* | 67.36 ± 6.15 | **68.43 ± 5.43** | 66.98 ± 4.14 | 65.03 ± 5.28 | 65.41 ± 5.11 |
| auto10 | 37.24 ± 4.53 | *55.71 ± 4.88* | 55.10 ± 4.99 | 51.36 ± 11.00 | 52.72 ± 10.56 | 47.31 ± 4.86 | **56.12 ± 4.48** |
| housing10 | 35.96 ± 2.96 | 58.01 ± 5.18 | *58.32 ± 3.93* | 56.33 ± 4.96 | 56.51 ± 4.02 | 55.30 ± 4.23 | **59.00 ± 3.97** |
| stock10 | 34.10 ± 3.02 | 68.45 ± 2.42 | 74.85 ± 3.21 | **82.31 ± 2.13** | *82.28 ± 2.10* | 71.76 ± 3.57 | 79.08 ± 2.37 |
| Ranking | 5.71 | 4.46 | 3.36 | **3.04** | 3.29 | 4.95 | *3.20* |
| Friedman's test: Confidence interval $C_0 = (0, F_{\alpha=0.05}) = 2.15$, F-val$_{Acc}$ : $8.24 \notin C_0$ | | | | | | | |

The best method is in **bold** face and the second one in *italics*.

Table 5: Results obtained for each method reported in terms of *MAE*.

| Dataset | POM | K-POM | KRGB-POM | KLRGB-POM | OKLRGB-POM | KDLOR | SVORIM |
|---|---|---|---|---|---|---|---|
| contact-lenses | 0.500 ± 0.255 | 0.722 ± 0.298 | 0.561 ± 0.257 | *0.378 ± 0.138* | **0.377 ± 0.138** | 0.656 ± 0.239 | 0.522 ± 0.122 |
| pasture | 0.589 ± 0.168 | 0.426 ± 0.168 | 0.370 ± 0.132 | 0.356 ± 0.132 | *0.348 ± 0.116* | 0.396 ± 0.116 | **0.333 ± 0.113** |
| squash-stored | 0.792 ± 0.260 | 0.426 ± 0.153 | **0.308 ± 0.128** | *0.344 ± 0.121* | 0.351 ± 0.117 | 0.362 ± 0.147 | 0.377 ± 0.118 |
| squash-unstored | 0.797 ± 0.246 | 0.385 ± 0.146 | 0.238 ± 0.111 | **0.221 ± 0.121** | *0.231 ± 0.123* | 0.267 ± 0.138 | 0.246 ± 0.120 |
| tae | 0.751 ± 0.200 | 0.650 ± 0.063 | 0.533 ± 0.106 | *0.456 ± 0.065* | 0.465 ± 0.067 | **0.453 ± 0.058** | 0.468 ± 0.071 |
| SWD | 0.504 ± 0.035 | 0.460 ± 0.029 | *0.446 ± 0.037* | **0.445 ± 0.035** | 0.448 ± 0.037 | 0.591 ± 0.033 | *0.446 ± 0.029* |
| diabetes5 | 0.670 ± 0.130 | 0.733 ± 0.151 | 0.718 ± 0.272 | 0.633 ± 0.179 | **0.620 ± 0.143** | *0.621 ± 0.093* | 0.667 ± 0.099 |
| pyrim5 | 0.711 ± 0.155 | **0.442 ± 0.114** | 0.474 ± 0.129 | 0.470 ± 0.110 | 0.465 ± 0.115 | 0.596 ± 0.124 | *0.449 ± 0.125* |
| triazines5 | 1.053 ± 0.032 | *0.677 ± 0.049* | 0.842 ± 0.428 | 0.738 ± 0.131 | 0.713 ± 0.080 | **0.671 ± 0.032** | *0.677 ± 0.035* |
| wisconsin5 | 1.144 ± 0.156 | **1.007 ± 0.088** | 1.401 ± 0.373 | 1.041 ± 0.051 | 1.043 ± 0.051 | 1.110 ± 0.022 | *1.040 ± 0.058* |
| machine5 | *0.178 ± 0.045* | 0.198 ± 0.052 | 0.195 ± 0.064 | 0.184 ± 0.063 | **0.177 ± 0.043** | 0.214 ± 0.062 | 0.181 ± 0.038 |
| toy | 0.944 ± 0.122 | 0.090 ± 0.034 | *0.079 ± 0.031* | 0.088 ± 0.031 | 0.092 ± 0.036 | 0.111 ± 0.032 | **0.052 ± 0.026** |
| auto5 | 0.385 ± 0.054 | **0.246 ± 0.040** | *0.251 ± 0.043* | 0.255 ± 0.050 | 0.265 ± 0.053 | 0.297 ± 0.051 | 0.262 ± 0.036 |
| housing5 | 0.404 ± 0.041 | 0.283 ± 0.045 | 0.270 ± 0.045 | *0.250 ± 0.031* | **0.243 ± 0.033** | 0.269 ± 0.032 | 0.251 ± 0.034 |
| eucalyptus | 1.940 ± 0.276 | 0.557 ± 0.045 | 0.529 ± 0.051 | **0.436 ± 0.031** | 0.453 ± 0.142 | 0.504 ± 0.046 | *0.439 ± 0.032* |
| stock5 | 0.375 ± 0.022 | 0.158 ± 0.019 | 0.124 ± 0.017 | *0.111 ± 0.017* | **0.110 ± 0.020** | 0.148 ± 0.021 | 0.121 ± 0.018 |
| LEV | 0.505 ± 0.033 | 0.418 ± 0.029 | 0.416 ± 0.024 | *0.413 ± 0.031* | **0.412 ± 0.030** | 0.514 ± 0.034 | 0.420 ± 0.030 |
| automobile | 1.153 ± 0.750 | 0.522 ± 0.072 | 0.411 ± 0.101 | 0.402 ± 0.073 | 0.393 ± 0.076 | *0.384 ± 0.088* | **0.368 ± 0.075** |
| heating | 0.555 ± 0.040 | 0.341 ± 0.032 | **0.134 ± 0.028** | 0.200 ± 0.058 | *0.184 ± 0.022* | 0.225 ± 0.067 | 0.273 ± 0.065 |
| cooling | 0.580 ± 0.049 | 0.396 ± 0.026 | **0.272 ± 0.031** | 0.307 ± 0.028 | 0.305 ± 0.030 | *0.296 ± 0.036* | 0.350 ± 0.062 |
| diabetes10 | 1.442 ± 0.331 | 1.645 ± 0.296 | 1.500 ± 0.351 | *1.382 ± 0.328* | **1.352 ± 0.222** | 1.521 ± 0.256 | 1.527 ± 0.291 |
| pyrim10 | 1.344 ± 0.214 | *1.058 ± 0.196* | 1.351 ± 0.429 | 1.379 ± 0.163 | 1.332 ± 0.193 | 1.342 ± 0.274 | **0.995 ± 0.185** |
| triazines10 | 2.742 ± 0.417 | *1.311 ± 0.095* | 2.188 ± 1.363 | 1.409 ± 0.426 | 1.548 ± 0.654 | 1.438 ± 0.081 | **1.288 ± 0.095** |
| wisconsin10 | 2.431 ± 0.190 | **2.224 ± 0.139** | 3.228 ± 0.780 | 2.251 ± 0.124 | *2.232 ± 0.087* | 2.359 ± 0.051 | 2.319 ± 0.099 |
| machine10 | 0.534 ± 0.137 | 0.501 ± 0.132 | 0.516 ± 0.142 | **0.451 ± 0.099** | *0.464 ± 0.081* | 0.531 ± 0.146 | 0.482 ± 0.109 |
| auto10 | 0.769 ± 0.070 | *0.518 ± 0.058* | 0.525 ± 0.067 | 0.645 ± 0.412 | 0.610 ± 0.399 | 0.680 ± 0.075 | **0.504 ± 0.057** |
| housing10 | 0.844 ± 0.061 | 0.525 ± 0.073 | *0.502 ± 0.058* | 0.562 ± 0.099 | 0.580 ± 0.074 | 0.521 ± 0.056 | **0.482 ± 0.059** |
| stock10 | 0.870 ± 0.049 | 0.319 ± 0.025 | 0.258 ± 0.032 | *0.180 ± 0.021* | **0.179 ± 0.021** | 0.290 ± 0.035 | 0.212 ± 0.024 |
| Ranking | 6.25 | 4.52 | 4.09 | 2.93 | **2.71** | 4.54 | *2.96* |
| Friedman's test: Confidence interval $C_0 = (0, F_{\alpha=0.05}) = 2.15$, F-val$_{Acc}$ : $13.86 \notin C_0$ | | | | | | | |

The best method is in **bold** face and the second one in *italics*.

capability of the proposals to deal with nonlinearly separable data. Indeed, in those datasets where the POM has achieved better results, the proposed methods also obtained a similar performance.

As can be observed in the results, OKLRGB-POM performs better than KLRGB-POM in *MAE* but worse in *Acc*. This is a consequence of the cost matrices introduced in Section 3.2, where a uniform cost for all errors (KLRGB) is clearly favour-

ing accuracy, while non-uniform costs (OKLRGB) are better for reducing *MAE*. However, when dealing with ordinal regression problems, classifiers obtaining better *MAE* results are generally preferred. Similar conclusions were found in [29] when comparing the results obtained by SVORIM to its explicit version, SVOREX (where only adjacent classes are taken into account for the slacks).

To determine the statistical significance of the differences observed between the different methodologies, a procedure to compare multiple classifiers in multiple datasets is employed [30]. Table 4 and Table 5 also show the result of applying the statistical non-parametric Friedman's test (for a significance level of and $\alpha = 0.05$) to the mean *Acc* and *MAE* rankings. It can be seen that the test rejects the null-hypothesis that all of the algorithms perform similarly in mean ranking for all the metrics (note that for *MAE* the significant differences are larger).

On the basis of this rejection and following the guidelines in [30], we consider the best performing methods in *Acc* and *MAE* (*i.e.*, KLRGB-POM and OKLRGB-POM, respectively) as control methods for the following tests. Furthermore, we also consider the method POM as a control method, to analyse the performance of the original linear method with respect to the rest of developed techniques. We compare these three methods to the rest according to their rankings. The Holm's test is an approach to compare all classifiers to a given classifier (a control method). The test statistics for comparing the $i$-th and $j$-th method using this procedure is:

$$z = \frac{R_i - R_l}{\sqrt{\frac{L(L+1)}{6T}}},$$

where $L$ is the number of algorithms, $T$ is the number of datasets and $R_i$ is the mean ranking of the $i$-th method. The $z$ value is used to find the corresponding probability from the table of the normal distribution, which is then compared with an appropriate level of significance $\alpha$. Holm's test adjusts the value for $\alpha$ in order to compensate for multiple comparisons. This is done in a step-up procedure that sequentially tests the hypotheses ordered by their significance. We denote the ordered p-values by $p_1, p_2, \ldots, p_q$ so that $p_1 \leq p_2 \leq \ldots \leq p_q$. Holm's test compares each $p_i$ with $\alpha^*_{\text{Holm}} = \alpha/(L-i)$, starting from the most significant $p$ value. If $p_1$ is below $\alpha/(L-1)$, the corresponding hypothesis is rejected and we allow to compare $p_2$ with $\alpha/(L-2)$. If the second hypothesis is rejected, the test proceeds with the third, and so on.

This process is included in Table 6, where the results from the Holm statistical test are shown. Several conclusions can be drawn. First, it can be seen that the POM algorithm is significantly improved by most of the algorithms in terms of *Acc* and *MAE*, specially by KLRGB-POM in *Acc* and OKLRGB-POM in *MAE*. Considering the KLRGB-POM method, one can appreciate that it significantly outperforms the K-POM technique, meaning this that the use of the regularised gradient-based technique for selecting the optimal dimensions helps to improve the performance of the proposed kernelisation of the POM method. It is also better than the KDLOR method. However, no significant differences can be seen between this technique and the rest of methodologies that make use of this regularised gradient-based technique (although it presents better performance in mean ranking, as can be seen in Table 4 and Table 5). Concerning the ordinal version (OKLRGB-POM), similar results can be found, although in this case, there exists significant differences with respect to KRGB-POM in *MAE* (which is similar to KLRGB-POM but using gradient descent for adjusting the kernel). As can be seen, the developed techniques present statistically significant differences when compared to KDLOR, and improved SVORIM results (although not significantly). We should take into account that SVORIM is indeed one of the most successful and widely used technique in the state-of-the-art of ordinal regression [2]. As a summary, both multiple kernel proposals (OKLRGB-POM and KLRGB-POM) improve the results of POM and other kernel techniques (KDLOR and K-POM), while OKLRGB-POM is also able to improve the results from the proposal based on gradient descent (KRGB-POM). The kernelisation strategy is suitable for enabling the POM method to perform nonlinear decision boundaries and to reach the state-of-the-art results (SVORIM), while still obtaining natural probability estimations, which can only be approximated by POM.

We now analyse how the selection of the dimensions differ for all the datasets. This is done by considering K-POM and KRGB-POM methods, that make use of different strategies for selecting the dominant dimensions of the projected subspace and result in very different performance. Table 7 reports the percentage of agreement between both selections (i.e, if both algorithms consider the base $\mathbf{u}_i$ to be suitable, or the contrary). From this result, it can be seen that although from certain datasets the level of agreement is very high (meaning this that the selected dimensions for the KRGB-POM method are the ones associ-

Table 6: Results of the Holm procedure using POM, KLRGB-POM and OKLRGB-POM as control methods: corrected $\alpha$ values, compared method and $p$-values, ordered by the number of comparison ($i$).

| | Control alg.: POM | | *Acc* | | *MAE* | |
|---|---|---|---|---|---|---|
| $i$ | $\alpha^*_{0.05}$ | $\alpha^*_{0.10}$ | Method | $p_i$ | Method | $p_i$ |
| 1 | 0.0083 | 0.0167 | KLRGB-POM | $0.0000_{--}$ | OKLRGB-POM | $0.0000_{--}$ |
| 2 | 0.0100 | 0.0200 | SVORIM | $0.0000_{--}$ | KLRGB-POM | $0.0000_{--}$ |
| 3 | 0.0125 | 0.0250 | OKLRGB-POM | $0.0000_{--}$ | SVORIM | $0.0000_{--}$ |
| 4 | 0.0167 | 0.0333 | KRGB-POM | $0.0000_{--}$ | KRGB-POM | $0.0001_{--}$ |
| 5 | 0.0250 | 0.0500 | K-POM | $0.0304_{-}$ | K-POM | $0.0027_{--}$ |
| 6 | 0.0500 | 0.1000 | KDLOR | 0.1853 | KDLOR | $0.0029_{--}$ |
| | Control alg.: KLRGB-POM | | *Acc* | | *MAE* | |
| $i$ | $\alpha^*_{0.05}$ | $\alpha^*_{0.10}$ | Method | $p_i$ | Method | $p_i$ |
| 1 | 0.0083 | 0.0167 | POM | $0.0000_{++}$ | POM | $0.0000_{++}$ |
| 2 | 0.0100 | 0.0200 | KDLOR | $0.0009_{++}$ | KDLOR | $0.0053_{++}$ |
| 3 | 0.0125 | 0.0250 | K-POM | $0.0133_{+}$ | K-POM | $0.0059_{++}$ |
| 4 | 0.0167 | 0.0333 | KRGB-POM | 0.5777 | KRGB-POM | 0.0444 |
| 5 | 0.0250 | 0.0500 | OKLRGB-POM | 0.6650 | OKLRGB-POM | 0.7105 |
| 6 | 0.0500 | 0.1000 | SVORIM | 0.7807 | SVORIM | 0.9507 |
| | Control alg.: OKLRGB-POM | | *Acc* | | *MAE* | |
| $i$ | $\alpha^*_{0.05}$ | $\alpha^*_{0.10}$ | Method | $p_i$ | Method | $p_i$ |
| 1 | 0.0083 | 0.0167 | POM | $0.0000_{++}$ | POM | $0.0000_{++}$ |
| 2 | 0.0100 | 0.0200 | KDLOR | $0.0040_{++}$ | KDLOR | $0.0016_{++}$ |
| 3 | 0.0125 | 0.0250 | K-POM | 0.0412 | K-POM | $0.0018_{++}$ |
| 4 | 0.0167 | 0.0333 | KLRGB-POM | 0.6650 | KRGB-POM | $0.0172_{+}$ |
| 5 | 0.0250 | 0.0500 | SVORIM | 0.8771 | SVORIM | 0.6650 |
| 6 | 0.0500 | 0.1000 | KRGB-POM | 0.9015 | KLRGB-POM | 0.7105 |

Win (++) or lose (−−) with statistical significant difference for $\alpha = 0.05$
Win (+) or lose (−) with statistical difference with $\alpha = 0.10$

ated to the first eigenvalues), for most of the datasets, the level of agreement is medium or low, indicating therefore that the selection of the most suitable dimensions is necessary.

Table 7: Agreement between the selected dimensions for K-POM and KRGB-POM methods.

| Dataset | Mean ± Std | Dataset | Mean ± Std |
|---|---|---|---|
| contact-lenses | 55.19 ± 21.49 | eucalyptus | 31.15 ± 10.16 |
| pasture | 64.07 ± 14.82 | stock5 | 68.15 ± 16.80 |
| squash-stored | 65.81 ± 21.83 | LEV | 84.67 ± 14.49 |
| squash-unstored | 68.63 ± 18.57 | automobile | 30.57 ± 7.39 |
| tae | 40.62 ± 11.88 | heating | 74.84 ± 6.65 |
| SWD | 83.37 ± 12.12 | cooling | 77.32 ± 8.39 |
| diabetes5 | 79.68 ± 17.22 | diabetes10 | 74.77 ± 16.26 |
| pyrim5 | 47.94 ± 19.93 | pyrim10 | 39.80 ± 13.90 |
| triazines5 | 19.51 ± 7.51 | triazines10 | 22.89 ± 11.10 |
| wisconsin5 | 10.05 ± 3.21 | wisconsin10 | 9.06 ± 3.87 |
| machine5 | 96.89 ± 0.96 | machine10 | 95.03 ± 0.83 |
| toy | 64.45 ± 15.23 | auto10 | 93.72 ± 4.86 |
| auto5 | 94.78 ± 4.10 | housing10 | 90.31 ± 0.77 |
| housing5 | 89.98 ± 6.22 | stock10 | 56.49 ± 23.05 |

## 5. Conclusions and future work

This paper explores the concept of the empirical feature space (an isomorphic space to the original feature space induced by the kernel trick) to reformulate a well-known ordinal regression method (the Proportional Odds Model or POM) in order to handle nonlinearly separable classification tasks. Different ideas are considered, such as the optimisation of the kernel matrix for tackling ordinal information and the optimisation of the dimensionality of the reduced empirical feature space. These proposals can be used to easily kernelise any existing liner ordinal regression method, independently of their formulation. The different experiments show that the proposed kernel techniques are able to increase the performance of linear ordinal regression methods, such as the POM and reach the performance of the state-of-the-art methods, while still being able to derive natural probability estimates. As future work, several promising lines can be introduced. Firstly, given the connection between our proposal and the Nyström approximation [31], we plan to reformulate this methodology in order to deal with large-scale datasets (by considering the steps followed for the Nyström method approximation). Note that our method, as it is at the moment, may be unaffordable for some large-scale problems, given the use of the singular value decomposition over the complete Gram matrix. Furthermore, because of the good synergy between the kernel learning technique and the proposed ordinal weight matrix, other ordinal kernel algorithms can also be used to analyse its performance.

## Acknowledgments

This work has been subsidized by the TIN2011-22794 project of the Spanish Ministerial Commission of Science and Technology (MICYT), FEDER funds and the P11-TIC-7508 project of the "Junta de Andalucía" (Spain). Manuel Cruz-Ramírez's research has been subsidized by the FPU Predoctoral Program (Spanish Ministry of Education and Science), grant reference AP2009-0487.

## References

[1] A. Agresti, Categorical Data Analysis, 2nd Edition, Wiley Series in Probability and Statistics, Wiley-Interscience, 2002.

[2] P. A. Gutiérrez, M. Pérez-Ortiz, F. Fernandez-Navarro, J. Sánchez-Monedero, C. Hervás-Martínez, An Experimental Study of Different Ordinal Regression Methods and Measures, in: 7th International Conference on Hybrid Artificial Intelligence Systems (HAIS), 2012, pp. 296–307.

[3] P. McCullagh, Regression models for ordinal data, Journal of the Royal Statistical Society 42 (2) (1980) 109–142.

[4] W. Chu, S. S. Keerthi, Support vector ordinal regression, Neural Computation 19 (2007) 792–815.

[5] B.-Y. Sun, J. Li, D. D. Wu, X.-M. Zhang, W.-B. Li, Kernel discriminant learning for ordinal regression, IEEE Transactions on Knowledge and Data Engineering 22 (2010) 906–910.

[6] J. S. Cardoso, J. F. P. da Costa, Learning to classify ordinal data: The data replication method, Journal of Machine Learning Research 8 (2007) 1393–1429.

[7] W.-Y. Deng, Q.-H. Zheng, S. Lian, L. Chen, X. Wang, Ordinal extreme learning machine, Neurocomputing 74 (1–3) (2010) 447–456.

[8] B. Schölkopf, S. Mika, C. J. C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, A. J. Smola, Input space versus feature space in kernel-based methods, IEEE Transactions on Neural Networks 10 (1999) 1000–1017.

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

[10] S. Abe, K. Onishi, Sparse least squares support vector regressors trained in the reduced empirical feature space, in: Proc. of the 17th international conference on Artificial neural networks, ICANN, Springer-Verlag, 2007, pp. 527–536.

[11] H. Xiong, A unified framework for kernelization: The empirical kernel feature space, in: Chinese Conference on Pattern Recognition (CCPR), 2009, pp. 1 –5.

[12] C. Cortes, M. Mohri, A. Rostamizadeh, Algorithms for learning kernels based on centered alignment, Journal of Machine Learning Research 13 (2012) 795–828.

[13] N. Cristianini, J. Kandola, A. Elisseeff, J. Shawe-Taylor, On kernel-target alignment, in: Advances in Neural Information Processing Systems 14, MIT Press, 2002, pp. 367–373.

[14] J. Verwaeren, W. Waegeman, B. De Baets, Learning partial ordinal class memberships with kernel-based proportional odds models, Comput. Stat. Data Anal. 56 (4) (2012) 928–942.

[15] G. S. Kimeldorf, G. Wahba, Some results on Tchebycheffian spline functions, Journal of Mathematical Analysis and Applications 33 (1) (1971) 82–95.

[16] M. J. Mathieson, Ordinal models for neural networks, in: J. M. A.-P. N. Refenes, Y. Abu-Mostafa, A. Weigend (Eds.), Proceedings of the Third International Conference on Neural Networks in the Capital Markets, Neural Networks in Financial Engineering, World Scientific, 1996, pp. 523–536.

[17] M. Ramona, G. Richard, B. David, Multiclass feature selection with kernel gram-matrix-based criteria, IEEE Trans. Neural Netw. Learning Syst. 23 (10) (2012) 1611–1623.

[18] B. Schölkopf, A. Smola, K.-R. Müller, Nonlinear component analysis as a kernel eigenvalue problem, Neural Computation 10 (5) (1998) 460–474.

[19] J. Zhu, T. Hastie, Kernel logistic regression and the import vector machine, Journal of Computational and Graphical Statistics 14 (1) (2001) 185–205.

[20] Y. Guermeur, A. Lifchitz, R. Vert, Kernel for protein secondary structure prediction, The MIT Press, Cambridge, Massachussetts, 2004, p. 416.

[21] S. Baccianella, A. Esuli, F. Sebastiani, Evaluation measures for ordinal regression, in: Proceedings of the Ninth International Conference on Intelligent Systems Design and Applications (ISDA 09), Pisa, Italy.

[22] M. Cruz-Ramírez, C. Hervás-Martínez, J. Sánchez-Monedero, P. Gutiérrez, Metrics to guide a multi-objective evolutionary algorithm for ordinal classification, Neurocomputing 135 (2014) 21–31, advances in Learning Schemes for Function Approximation, Selected

papers from the 11th International Conference on Intelligent Systems Design and Applications (ISDA 2011).
[23] R. Tibshirani, Regression shrinkage and selection via the lasso, Journal of the Royal Statistical Society. Series B (Methodological) (1996) 267–288.
[24] C. Igel, M. Hüsken, Empirical evaluation of the improved rprop learning algorithms., Neurocomputing 50 (2003) 105–123.
[25] T. Glasmachers, Gradient based optimization of support vector machines, Ph.D. thesis (2008).
[26] A. Asuncion, D. Newman, UCI machine learning repository (2007). URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`
[27] PASCAL, Pascal (pattern analysis, statistical modelling and computational learning) machine learning benchmarks repository (2011). URL `http://mldata.org/`
[28] W. Chu, Z. Ghahramani, Gaussian processes for ordinal regression, Journal of Machine Learning Research 6 (2005) 1019–1041.
[29] W. Chu, S. S. Keerthi, Support vector ordinal regression, Neural Computation 19 (3) (2007) 792–815.
[30] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research 7 (2006) 1–30.
[31] P. Drineas, M. W. Mahoney, On the Nyström method for approximating a gram matrix for improved kernel-based learning, Journal of Machine Learning Research 6 (2005) 2153–2175.

María Pérez-Ortiz was born in Córdoba, Spain. She received her B.S. degree in Computer Science from the University of Córdoba, Spain, in 2011 and her M.Sc. degree in Intelligent Systems from the University of Córdoba, Spain, in 2012. She is currently working toward her Ph.D. degree in the Department of Computer Science and Numerical Analysis (University of Córdoba, Spain), in the area of computer science and artificial intelligence. Her current interests include a wide range of topics concerning machine learning and pattern recognition.

Pedro Antonio Gutiérrez was born in Córdoba, Spain. He received the B.S. degree in Computer Science from the University of Sevilla, Spain, in 2006, and the Ph.D. degree in Computer Science and Artificial Intelligence from the University of Granada, Spain, in 2009. He is currently an Assistant Professor in the Department of Computer Science and Numerical Analysis, University of Córdoba. His current research interests include pattern recognition, neural networks, evolutionary computation and their application to real world problems.

Manuel Cruz-Ramírez was born in Córdoba, Spain. He received the B.S. degree in Computer Science from the University of Córdoba, Spain, in 2009, the M.S. in Soft Computing and Intelligent Systems from the University of Granada, Spain, in 2010 and the Ph.D degree in the official Ph.D Program on Information and Communication Technologies from the University of Granada, Spain, in 2013. He is currently working at the Department of Computer Science and Numerical Analysis, University of Córdoba, Spain. His current research interests include neural networks, multi-objective evolutionary algorithms and their applications to real problems and the optimisation of performance/evaluation measures.

Javier Sánchez-Monedero was born in Córdoba (Spain). He received the B.S in Computer Science from the University of Granada, Spain, in 2008 and the M.S. in Multimedia Systems from the University of Granada in 2009. In 2013 he obtained the Ph.D. degree on Information and Communication Technologies of the University of Granada. He is working as researcher with the Department of Computer Science and Numerical Analysis at University of Córdoba. His current research interests include computational intelligence methods and their applications, as well as distributed systems.

César Hervás-Martínez was born in Cuenca, Spain. He received the B.S. degree in Statistics and Operating Research from the Universidad Complutense, Madrid, Spain, in 1978, and the Ph.D. degree in mathematics from the University of Seville, Seville, Spain, in 1986. He is currently a Professor with the Department of Computing and Numerical Analysis, University of
Córdoba, Córdoba, Spain, in the area of computer science and artificial intelligence and an Associate Professor with the Department of Quantitative Methods, School of Economics. His current research interests include learning algorithms, neural networks, pattern recognition and the modeling of natural systems.