

---

# Bayesian Sparse Factor Analysis with Kernelized Observations

---

Carlos Sevilla-Salcedo,\* Alejandro Guerrero-López, Pablo M. Olmos,† Vanessa Gómez-Verdejo

Department of Signal Processing and Communications, Universidad Carlos III de Madrid Leganés, 28911 Spain

## Abstract

Multi-view problems can be faced with latent variable models since they are able to find low-dimensional projections that fairly capture the correlations among the multiple views that characterise each datum. On the other hand, high-dimensionality and non-linear issues are traditionally handled by kernel methods, inducing a (non)-linear function between the latent projection and the data itself. However, they usually come with scalability issues and exposition to overfitting. Here, we propose merging both approaches into single model so that we can exploit the best features of multi-view latent models and kernel methods and, moreover, overcome their limitations.

In particular, we combine probabilistic factor analysis with what we refer to as kernelized observations, in which the model focuses on reconstructing not the data itself, but its relationship with other data points measured by a kernel function. This model can combine several types of views (kernelized or not), and it can handle heterogeneous data and work in semi-supervised settings. Additionally, by including adequate priors, it can provide compact solutions for the kernelized observations -based in a automatic selection of Bayesian Relevance Vectors (RVs)- and can include feature selection capabilities. Using several public databases, we demonstrate the potential of our approach (and its extensions) w.r.t. common multi-view learning models such as kernel canonical correlation analysis or manifold relevance determination.

## 1 Introduction

Given a set of observable data, Latent Variable Models (LVMs) [1] aim to extract a reduced set of hidden variables able to summarise the information into a low dimensional space. These models have become crucial in multi-view problems [2, 3, 4], where data is represented by different modalities or views, since LVMs are able to explain the common information among all the modalities. One of the most well-known types of LVMs are the classical MultiVariate Analysis (MVA) methods, such as Principal Component Analysis (PCA) and Canonical Correlation Analysis (CCA) [5, 6], which aim to exploit the data correlation to obtain a low dimensional latent representation of the data. Its usage has been generalised due to its easy non-linear extension by means of kernel methods [7, 8]. The fact of supporting a kernel formulation allows these methods to learn arbitrarily complex non-linear models with a complexity determined by the number of training points [9] and make them highly convenient in scenarios with high dimensional data.

Probabilistic formulations of LVMs methods are known as Factor Analysis (FA) [10], which emerge as a linear Bayesian framework where one can obtain the desired latent representation together with a measure of the uncertainty. Among their many variants, such as Probabilistic PCA [11], Supervised

---

\*Corresponding author. Email address: sevisal@tsc.uc3m.es

†Pablo M. Olmos is also with the Gregorio Marañón Health Research Institute.

PCA [12], Bayesian Factor Regression [13] or Bayesian CCA [14], the Inter-Battery FA model [14] stands out for its capability of handling not only latent variables associated to the common information among all the views, but also for being able to model the intra-view information. This model has been recently extended in [15], named as Sparse Semi-supervised Inter-Battery Bayesian Analysis (SSHIBA), to incorporate missing attributes, feature selection and the ability to handle heterogeneous data such as categorical or multi-dimensional binary data.

The use of kernel methods in Bayesian approaches has been mostly developed with Gaussian Processes (GP) [16] and their non-supervised version to perform dimensionality reduction explored in GPLVMs [17]. These approaches combine the advantages of the kernels methods, exploiting the non-linear relationships among the data, with those of a probabilistic framework. In [18], the authors propose a shared GPLVMs approach, called Manifold Relevance Determination (MRD), to provide a non-linear latent representation for multi-view learning problems. This model is extended in [19], including an Automatic Relevance Determination (ARD) prior [20] over the kernel formulation, to endow it with feature relevance analysis.

GPLVMs come with practical scalability drawbacks that need to be addressed. The cubic complexity with the number of training points requires the use of inducing points and variational approaches [21]. Selecting the number of inducing points to use, and where to place them in the latent space, is still a challenging problem, being a common solution to place them in a regular basis along the latent space and only optimize the pseudo-observation at those points [22]. Furthermore, up to our knowledge, there is no versatile implementation in the state-of-the-art of a multi-view GPLVM able to handle heterogeneous observations (integer, categorical, real and positive observations) and missing values.

In this paper we propose a novel method to implement non-linear probabilistic LVMs that still builds upon a linear generative model, hence inheriting their computational and scalability properties. Instead of implementing a kernel method, i.e. a GP, to move from the latent representation to the observed data, we propose to reformulate probabilistic FA so that it generates kernel relationships instead of data observations. In the same way that Kernelized PCA (KPCA) or Kernelized CCA (KCCA) are able to generate non-linear latent variables by linearly combining element of a kernel vector; here, from a Bayesian generative point of view, we first i.i.d. sample latent representations and project on an  $N$ -dimensional space (being  $N$  the number of points) using a weight matrix representing the dual parameters. We apply this trick over the SSHIBA formulation [15] to exploit their functionalities over this kernelized formulation. Thanks to that, we can efficiently face semi-supervised heterogeneous multi-view problems combining linear and non-linear data representations; in this way, one can combine kernelized views to deal with non-linear relationships with linearly kernelized to work with high dimensional problems. Although [23] presents a first attempt of using kernelized data representations for FA models, here we propose a complete framework including the following novelties/functionalities: (1) we can work with any type of kernel and are not limited to linear kernels; (2) we can force the automatic selection of Relevance Vectors (RVs) of the kernel to obtain scalable solutions (equivalent to the support vectors of the Support Vector Machines (SVMs)); (3) we can include an ARD kernel to obtain a per-variable relevance that can be exploited for feature selection; (4) we can adapt the multiview model to work as multiple kernel learning algorithm for FA models.

The article is organised as follows: Section 2 presents the kernelized formulation of SSHIBA as well as the proposed formulations for the RVs and feature selection. Section 3 defines the implementation details and the baselines and databases used for the experiments in different scenarios, where we show the performance of each proposed extension to the model. Finally, Section 4 gives some final conclusions and highlights the main results.

## 2 Bayesian Sparse Factor Analysis with Kernelized Observations

Let's consider a multi-view problem where we have  $N$  data samples represented in  $M$  different modalities,  $\{\mathbf{X}^{(m)}\}_{m=1}^M$ , and our goal is to find an inter and intra-view non-linear latent representation,  $\mathbf{Z}$ . That is, given that  $\mathbf{x}_{n,:}^{(m)} \in \mathbb{R}^{D_m}$  is the  $m$ -th view of the  $n$ -th data,  $\mathbf{z}_{n,:}$  has to compress, in a low dimensional space of size  $K_c \ll (D_1, \dots, D_M)$  both the intra-view and inter-view information of  $\mathbf{x}_{n,:}^{(m)}$  over all the views exploiting the correlations among the data.<sup>1</sup> Whereas kernel MVA obtains the

<sup>1</sup>Given a matrix  $\mathbf{B}$ , we denote the  $i$ -th row by  $\mathbf{b}_{i,:}$  and the  $j$ -th column by  $\mathbf{b}_{:,j}$ .

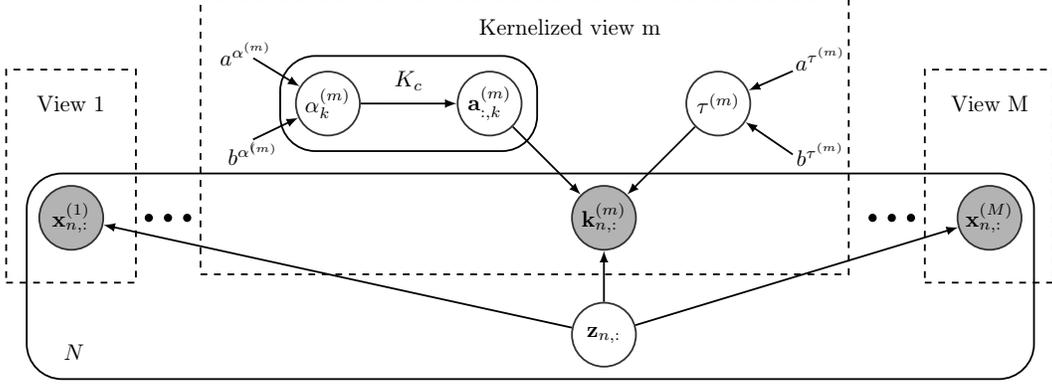


Figure 1: Graphic model of SSHIBA with kernelized views (KSSHIBA).

latent representation of the  $n$ -th data as a linear combination, by some dual variables, of its kernel representation  $\mathbf{k}_{n,:}^{(m)}$ , here we propose to reformulate this idea from a generative point of view. In particular, we start from the SSHIBA algorithm formulation [15] (see A for a brief introduction) and consider that there exist some latent variables  $\mathbf{z}_{n,:} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{K_c})$  which are linearly combined with a set of dual variables  $\mathbf{A}^{(m)} \in \mathbb{R}^{N \times K_c}$  to generate a kernel vector,  $\mathbf{k}_{n,:}^{(m)}$ , as:

$$\mathbf{k}_{n,:}^{(m)} = \mathbf{z}_{n,:} \mathbf{A}^{(m)\top} + \tau^{(m)} \quad (1)$$

where  $\tau^{(m)}$  is zero-mean Gaussian noise, with noise power following a Gamma distribution of parameters  $a^{\tau^{(m)}}$  and  $b^{\tau^{(m)}}$ , and, given a mapping function  $\phi(\cdot)$  and its associated kernel function  $K(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^\top \phi(\mathbf{x}')$ ,  $\mathbf{k}_{n,:}^{(m)}$  is the kernel between  $\mathbf{x}_{n,:}^{(m)}$  and all the training data, i.e.,  $\mathbf{k}_{n,:}^{(m)} = [K(\mathbf{x}_{n,:}^{(m)}, \mathbf{x}_{1,:}^{(m)}), \dots, K(\mathbf{x}_{n,:}^{(m)}, \mathbf{x}_{N,:}^{(m)})]$ . The dual variable matrix  $\mathbf{A}^{(m)}$  plays the role of the linear projection matrix and it is defined using the same structured ARD prior considered in both [14] and [15]. Namely, an ARD prior that promotes that full rows of this matrix are cancelled, i.e.

$$\mathbf{a}_{:,k}^{(m)} \sim \mathcal{N}\left(0, \left(\alpha_k^{(m)}\right)^{-1} I_{K_c}\right) \quad (2)$$

$$\alpha_k^{(m)} \sim \Gamma\left(a^{\alpha^{(m)}}, b^{\alpha^{(m)}}\right), \quad (3)$$

so that the product in (1) induces sparsity in the latent factors, leading to a selection of the appropriate set of them [24].

Figure 1 shows the graphical model of KSSHIBA. Following [14], for the data views that are directly explained given the latent projection we have  $\mathbf{x}_{n,:}^{(m)} = \mathbf{z}_{n,:} \mathbf{W}^{(m)\top} + \tau^{(m)}$ , where the weight matrix  $\mathbf{W}^{(m)}$  follows the same structured ARD prior mentioned above. We can refer to these as primal observations. For some other views, we might be interested in explaining them indirectly through a kernelized observation following (1). This conversion can be of interest when either we want to exploit intra and inter-view non-linear relationships, or when the view's dimensionality is much larger than the number of data points  $N$  and we prefer to work in the dual space, to reduce the number of parameters to be learnt. When both primal and kernelized observations are used, the learned latent projection  $\mathbf{z}_{n,:}$  attempts to faithfully reconstruct the original data representation in the primal views, as well as their kernel representation, i.e., their (non)-linear relationships with the other data points.

Note that sampling from the model in (1) does not ensure a valid kernel positive semi-definite matrix. The kernel matrix is simply treated as an observation (a kernelized observation) and, as such, the model parameters will be chosen to minimize the reconstruction error. In Figure 2 we include a graphical representation of both a kernelized observation and the map reconstruction through (1) using the mean of the posterior distribution of  $\mathbf{z}_{n,:}$ , where we can observe that the kernelized observations (or kernel matrix) are almost perfectly reconstructed. Certainly, more appropriate models could be used to adapt the observation model (given  $\mathbf{z}_{n,:}$ ) to the properties of a kernelized observation. To

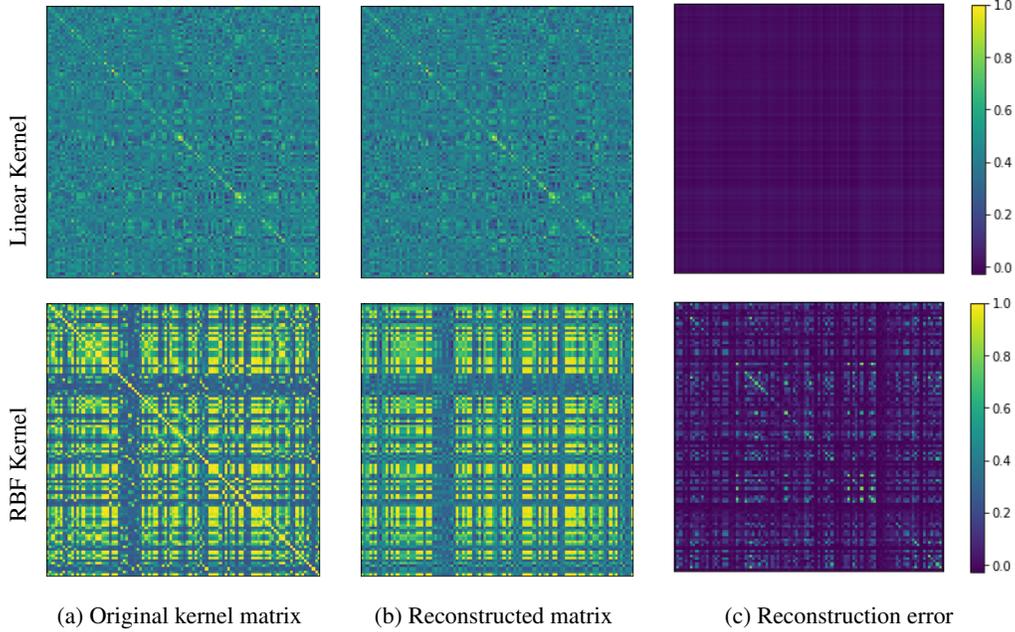


Figure 2: Example of the generative properties of KSSHIBA to reconstruct a complete kernel matrix.

address this issue, we have explored alternative formulations based in non-independent noise; for example, defining the noise distribution as an inverse-wishart to have a full rank covariance noise or modelling its covariance as the product of two low rank matrices [25]. However, these schemes led to considerably more complicated (less flexible) formulation which limited the rest of the properties of this proposal (as the ones proposed in the following sections). Henceforth, in this work, we restrict to the model in (1), and leave this line of work open for future research.

With this model we can evaluate the posterior distribution of the variables given the observed data

$$p(\Theta | \mathbf{k}_{1,:}, \dots, \mathbf{k}_{N,:}) = \frac{\prod_{n=1}^N p(\mathbf{k}_{n,:} | \Theta) p(\Theta)}{p(\mathbf{k}_{1,:}, \dots, \mathbf{k}_{N,:})}, \quad (4)$$

$$p(\mathbf{k}_{1,:}, \dots, \mathbf{k}_{N,:}) = \int p(\Theta, \mathbf{k}_{1,:}, \dots, \mathbf{k}_{N,:}) d(\Theta), \quad (5)$$

where  $\Theta$  comprises all random variables (rv) in the model. As presented in [15], we rely on an approximate inference approach through mean-field variational inference [26], where a lower bound to (5) of the form

$$\log p(\mathbf{k}_{1,:}, \dots, \mathbf{k}_{N,:}) \geq \int q(\Theta) \log \left( \frac{\prod_{n=1}^N p(\mathbf{k}_{n,:} | \Theta) p(\Theta)}{q(\Theta)} \right) d(\Theta) \quad (6)$$

is maximised, and a fully factorised variational family is chosen to approximate the posterior distribution in (4)

$$p(\Theta | \mathbf{K}^{\{\mathcal{M}\}}) \approx \prod_{m=1}^{\mathcal{M}} \left( q(\mathbf{A}^{(m)}) q(\tau^{(m)}) \prod_{k=1}^{K_c} q(\alpha_k^{(m)}) \right) \prod_{n=1}^N q(\mathbf{z}_{n,:}) \quad (7)$$

where  $\mathbf{K}^{\{\mathcal{M}\}}$  represents a stacked version of the kernel for each sample,  $\mathbf{k}_{n,:}^{\{\mathcal{M}\}}$ , of dimension  $N \times D_m$ .

The mean-field posterior structure along with the lowerbound in (6) results into a feasible coordinate-ascent-like optimization algorithm in which the optimal maximization of each of the factors in (7) can be computed if the rest remain fixed using the following expression

$$q^*(\theta_i) \propto \mathbb{E}_{\Theta_{-i}} [\log p(\Theta, \mathbf{k}_{1,:}, \dots, \mathbf{k}_{N,:})], \quad (8)$$

where  $\Theta_{-i}$  comprises all rv but  $\theta_i$ . This new formulation is in general feasible since it does not require to completely marginalize  $\Theta$  from the joint distribution.

Redefining the input matrix as a kernel matrix leads to a formulation equivalent to SSHIBA, which, accordingly, permits to keep the previous model functionalities. Table 1 shows the KSSHIBA mean-field factor update rules. For a compact notation, we stuck in matrix  $\mathbf{Z}$ , of dimension  $N \times K_c$ , the latent projection of all data points and  $\langle \cdot \rangle$  represents the mean value of the rv. The posterior distribution of all model parameters and latent projections is approximated using variational inference with a fully factorized posterior where it can be noted that each update has a computational cost of  $O(N^2 K_c + K_c^3)$ .

Table 1: Updated rules, obtained by a mean field approximation, of  $q$  distribution for the different variables of KSSHIBA model.

Variable	$q^*$ distribution	Parameters
$\mathbf{z}_{n,:}$	$\mathcal{N}_{\mathbf{z}_{n,:}}   \mu_{\mathbf{z}_{n,:}}, \Sigma_{\mathbf{Z}}$	$\mu_{\mathbf{z}_{n,:}} = \sum_{m=1}^M \langle \tau^{(m)} \rangle \mathbf{K}^{(m)} \langle \mathbf{A}^{(m)} \rangle \Sigma_{\mathbf{Z}}$ $\Sigma_{\mathbf{Z}}^{-1} = \left( I + \sum_{m=1}^M \left( \langle \tau^{(m)} \rangle \langle \mathbf{A}^{(m)\top} \mathbf{A}^{(m)} \rangle \right) \right)$
$\mathbf{A}^{(m)}$	$\prod_{n=1}^N \left( \mathcal{N} \left( \mathbf{a}_{n,:}^{(m)}   \mu_{\mathbf{a}_{n,:}^{(m)}}, \Sigma_{\mathbf{A}^{(m)}} \right) \right)$	$\mu_{\mathbf{a}_{n,:}^{(m)}} = \langle \tau^{(m)} \rangle \mathbf{K}^{(m)\top} \langle \mathbf{Z} \rangle \Sigma_{\mathbf{A}^{(m)}}$ $\Sigma_{\mathbf{A}^{(m)}}^{-1} = \left( \text{diag}(\langle \alpha^{(m)} \rangle) + \langle \tau^{(m)} \rangle \langle \mathbf{Z}^\top \mathbf{Z} \rangle \right)$
$\alpha_k^{(m)}$	$\Gamma \left( \alpha_k^{(m)}   a_{\alpha_k^{(m)}}, b_{\alpha_k^{(m)}} \right)$	$a_{\alpha_k^{(m)}} = \frac{D_m}{2} + a^{\alpha^{(m)}}$ $b_{\alpha_k^{(m)}} = b^{\alpha^{(m)}} + \frac{1}{2} \langle \mathbf{A}^{(m)\top} \mathbf{A}^{(m)} \rangle_{k,k}$
$\tau^{(m)}$	$\Gamma \left( \tau^{(m)}   a_{\tau^{(m)}}, b_{\tau^{(m)}} \right)$	$a_{\tau^{(m)}} = \frac{D_m N}{2} + a^{\tau^{(m)}}$ $b_{\tau^{(m)}} = b^{\tau^{(m)}} + \frac{1}{2} \left( \sum_{n=1}^N \sum_{\tilde{n}=1}^{\tilde{N}} k_{n,\tilde{n}}^{(m)2} - 2 \text{Tr} \left\{ \langle \mathbf{A}^{(m)} \rangle \langle \mathbf{Z}^\top \rangle \mathbf{K}^{(m)} \right\} + \text{Tr} \left\{ \langle \mathbf{A}^{(m)\top} \mathbf{A}^{(m)} \rangle \langle \mathbf{Z}^\top \mathbf{Z} \rangle \right\} \right)$

## 2.1 Multiple kernel learning

By using different kernels in different views, the model adapts to a multiple kernel learning scenario (MKL SSHIBA). To further analyse the implications of this combination, let's analyse the update rule for the mean of latent variables  $\mathbf{Z}$

$$\mu_{\mathbf{z}_{n,:}} = \sum_{m=1}^M \langle \tau^{(m)} \rangle \mathbf{K}^{(m)} \langle \mathbf{A}^{(m)} \rangle \Sigma_{\mathbf{Z}}. \quad (9)$$

Here we can see that the latent factors are obtained as a linear combination of the kernels on each view. This entails the additional advantage of automatically learning these mixing coefficients of this combination (depending on parameters  $\tau^{(m)}$ ,  $\mathbf{A}^{(m)}$  and  $\Sigma_{\mathbf{Z}}$ ) through variational inference. Furthermore, if we define one view as an output view, the values that need to be predicted are calculated using this learnt variable  $\mathbf{Z}$ , therefore conserving the MKL structure for the prediction.

Most MKL methods are based on heuristics [27, 28] or two step optimization process [29]. However, the proposed model determines the kernel parameters through the mean-field update. Furthermore, this formulation also benefits from the advantages of the SSHIBA formulation, namely semi-supervised learning, feature selection and RV selection.

## 2.2 Automatic Bayesian Relevance Vector Selection

On the basis of a full  $N \times N$  kernel, with a more structured ARD prior we can achieve not only the shrinkage of the number of effective latent factors, but also a more compact representation of the data by means of a reduced kernel matrix in which only a reduced set of RVs are kept.

For this purpose, the proposed formulation can introduce a double ARD prior over the dual variables  $\mathbf{A}^{(m)}$ ,

$$a_{n,k}^{(m)} \sim \mathcal{N}\left(0, \left(\gamma_n^{(m)} \alpha_k^{(m)}\right)^{-1}\right) \quad (10)$$

This way,  $\alpha_k^{(m)}$  continues forcing row-wise sparsity to automatically select the number of latent factors and, additionally,

$$\gamma_n^{(m)} \sim \Gamma\left(a\gamma^{(m)}, b\gamma^{(m)}\right) \quad (11)$$

induces column-wise sparsity, which automatically forces many of the RVs to not influence in the final model, consequently achieving a more compact solution. Table 2 shows the KSSHIBA mean-field factor update rules for the double ARD case.

Table 2: Updated  $q$  distribution for the automatic RV selection.

Variable	$q^*$ distribution	Parameters
$\mathbf{A}^{(m)}$	$\prod_{n=1}^N \mathcal{N}\left(\mathbf{a}_{n,:}^{(m)} \mid \mu_{\mathbf{a}_{n,:}^{(m)}}, \Sigma_{\mathbf{a}_{n,:}^{(m)}}\right)$	$\mu_{\mathbf{A}^{(m)}} = \langle \tau^{(m)} \rangle \mathbf{X}^{(m)\top} \langle \mathbf{Z} \rangle \Sigma_{\mathbf{W}^{(m)}}$ $\Sigma_{\mathbf{a}_{n,:}^{(m)}}^{-1} = \text{diag}(\langle \boldsymbol{\alpha}^{(m)} \rangle) \langle \gamma_n^{(m)} \rangle + \langle \tau^{(m)} \rangle \langle \mathbf{Z}^\top \mathbf{Z} \rangle$
$\gamma^{(m)}$	$\prod_{n=1}^N \Gamma\left(\gamma_n^{(m)} \mid a_{\gamma_n^{(m)}}, b_{\gamma_n^{(m)}}\right)$	$a_{\gamma_n^{(m)}} = \frac{K_c}{2} + a\gamma^{(m)}$ $b_{\gamma_n^{(m)}} = b\gamma^{(m)} + \frac{1}{2} \sum_{k=1}^{K_c} \langle \alpha_k^{(m)} \rangle \langle a_{n,k}^{(m)} a_{n,k}^{(m)} \rangle$

## 2.3 Automatic feature-relevance determination

Furthermore, we can additionally endow the proposed kernelized data representation with automatic feature relevance. If by using the double ARD structure we can cancel full rows or columns, equivalently, by using an ARD kernel we can perform this feature relevance determination. In the ARD kernel, each feature of the original observations is multiplied by a variable  $\lambda_d^{(m)}$  in the kernel definition. For example, for a Radial Basis Function (RBF) kernel,

$$k_{n,n}^{(m)} = \exp\left(-\sum_{d=1}^{D_m} \left(x_{n,d}^{(m)} - x_{n,d}^{(m)}\right)^2 \lambda_d^{(m)}\right), \quad (12)$$

we can optimise  $\boldsymbol{\lambda}^{(m)} = [\lambda_1^{(m)}, \dots, \lambda_{D_m}^{(m)}]$  by maximising the lower bound of our mean field approach given by direct optimisation over the variational lower bound. In our model, if the  $m$ -th view is kernelized then the only term in the lower bound affected by the ARD is the  $\mathbb{E}_q[\ln(p(\mathbf{K}^{(m)} \mid \Theta))]$ , having that (see [15] for further details on the lower bound):

$$LB = -\frac{\langle \tau^{(m)} \rangle}{2} \sum_{n=1}^N \sum_{u=1}^N \left( k_{n,u}^{(m)2} - 2k_{n,u}^{(m)} \langle \mathbf{a}_{u,:}^{(m)} \rangle \langle \mathbf{z}_{n,:}^\top \rangle + \langle \mathbf{a}_{u,:}^{(m)\top}, \mathbf{a}_{u,:}^{(m)} \rangle \langle \mathbf{z}_{n,:}^\top, \mathbf{z}_{n,:} \rangle \right) \quad (13)$$

We alternate between mean-field updates over the variational bound and direct maximization of (13) w.r.t.  $\boldsymbol{\lambda}^{(m)}$  using any gradient ascend method (we use Pytorch and Adam for such updates). Finally, by setting a threshold for  $\boldsymbol{\lambda}^{(m)}$ , the model is capable of automatically selecting the most relevant features while training.

### 3 Results

Throughout this section we define the different baselines as well as the implementation details for the analysis of the methods in terms of performance and interpretability of the inferred model parameters and latent projections. Furthermore, an exemplary notebook with the *Python* code of the proposed method is openly available in this github repository: [KSSHIBA](#).

#### 3.1 Experimental setup

In order to truthfully analyse the performance of the proposed framework, we decided to compare it with some state-of-the-art algorithms with similar capabilities. In particular we wanted to focus on some of the most relevant algorithms for FA. For the multi-dimensional regression datasets, we compare ourselves with KPCA, KCCA, MRD, Support Vector Regression (SVR) and MultiLayer Perceptron (MLP). Conversely, for the classification datasets, we compare with CCA and Support Vector Machine (SVM). We will now explain the specific parameters and formulations we used for these baselines.

As we proposed a kernel version of a FA model, we decided to compare to most common non-linear LVM methods, such as, KPCA and KCCA with a RBF kernel. For each of these models we explored 20 values of  $\gamma$  kernel hyperparameter in log-scale from  $[10^{-8}$  to  $10^{0.5}]$  divided by the number of tasks ( $C$ ). In order to carry out predictions KPCA was combined with Linear Regression (KPCA+LR), as KCCA can work in a supervised manner, we decided to use it on its own and with LR (KCCA+LR) for a complete comparative analysis. The number of latent factors has been fixed to the maximum possible,  $C$ , in KCCA and to those which explain 95% of the variance in KPCA.

Due to the equivalency to some of our functionalities, namely working with multiview data, carrying out RV selection and latent representations, we compared our model with the MRD [18] including an ARD for RVs selection. We used the available library in *Matlab* [18], setting the number of latents to twice the number of tasks ( $2 * C$ ). We used the RBF kernel with ARD and set the number of model optimisation iterations to 100 due to the long computational time required to train.

Another kernel model used as baseline is SVR, where we also used a RBF kernel, exploring the same values of  $\gamma$  as on KCCA/KPCA. The regularization parameter,  $\lambda$ , has also been validated, exploring 11 values in a logarithmic scale from  $[10^{-4}$  to  $10^4]$ .

Finally, we also wanted to compare our model with a MLP neural network in two different scenarios: (1) a MLP to force a bottleneck of dimension  $C$  in the hidden layers to have the same dimensionality reduction as KCCA. For this scenario, we validated two different configurations: (i) one hidden layer with  $C$  neurons and (ii) two hidden layer with  $C$  neurons and number of features ( $D_m$ ) neurons, respectively. (2) A second scenario without bottleneck is presented, validating between three configurations: (i) one hidden layer with 100 neurons, (ii) two hidden layers with 100 and 50 neurons, and (iii) three hidden layers with 100, 50 and 100 neurons.

Regarding KSSHIBA, we have used its semi-supervised capability to predict the output, using the test samples (without their targets) during the training and, later, predicting their labels with the mean of the posterior distribution. To determine the number of iterations of the inference process of KSSHIBA, we used a convergence criteria based on the evolution of the lower bound. In particular, we stop the algorithm either when  $mean(LB[-101 : -2]) > LB[-1](1 - 10^{-4})$ , where  $LB[-1]$  is the lower bound at the last iteration and  $mean(LB[-101 : -2])$  the mean value of the previous values of the lower bound, or when it reaches  $10^4$  iterations. The KSSHIBA models were randomly initialized 10 times, keeping the one with the best lower bound. KSSHIBA automatically prunes the latent factors using the ARD prior included in the projection matrix  $\mathbf{W}^{(m)}$ .

We calculated the hyperparameters of each model with a nested 10-folds Cross-Validation (CV). The outer CV is used to divide the dataset into training and test partitions, while the inner CV is in charge of validation and, therefore, it divides the training partition into a second training set and a validation set. This way we were able to estimate the performance of the whole framework and, additionally, validate the model parameters. We used the coefficient of determination (R2) to compare the performance of the different variations of the methods and to adjust the method hyperparameters, which were CV.

### 3.2 Performance evaluation of KSSHIBA for multi-dimensional regression

This section aims to analyse the performance of KSSHIBA for semi-supervised multi-dimensional regression in comparison with some state-of-the-art baselines. To do so, we used 9 multi-dimensional regression datasets from the *Mulan* repository [30, 31, 32], whose main properties are summarized in Table 3.

Table 3: Characteristic of the multi-task databases used in this work.

Database	Samples	Features	Tasks
<i>at1pd</i>	337	411	6
<i>at7pd</i>	296	411	6
<i>oes97</i>	334	263	16
<i>oes10</i>	403	298	16
<i>edm</i>	154	16	2
<i>jura</i>	359	15	3
<i>wq</i>	1,060	16	14
<i>enb</i>	768	8	2
<i>slump</i>	103	7	3

Table 4 shows the results obtained by the proposed model in two different scenarios (one in which the number of latent factors  $K_c$  is automatically learnt with the ARD prior and another in which we set  $K_c$  to  $C$ ) and we compare their results with those of the baselines, namely, KCCA, KPCA, MRD, SVR and MLP.

Table 4: Results on multitask databases of KSSHIBA and the baselines. The white subrow represents the mean and standard deviation of R2 score and the gray subrow the number of effective latent factors found.

	KSSHIBA	KSSHIBA $K_c = C$	MRD	KPCA + LR	KCCA + LR	SVR-RBF	MLP
<i>at1pd</i>	<b>0.79 ± 0.09</b> 46 ± 6	0.78 ± 0.09 6	0.67 ± 0.07 12	0.67 ± 0.12 22 ± 10	0.75 ± 0.11 6	0.01 ±0.05	0.77 ±0.11
<i>at7pd</i>	0.50 ± 0.18 14 ± 6	0.52 ± 0.13 6	0.48 ± 0.12 12	0.39 ± 0.19 21 ± 1	<b>0.57 ± 0.16</b> 6	0.01 ±0.03	0.35 ±0.69
<i>oes97</i>	<b>0.71 ± 0.10</b> 17 ± 6	0.69 ± 0.10 16	0.34 ± 0.07 32	0.45 ± 0.20 12 ± 7	0.36 ± 0.09 16	0.39 ±0.10	0.58 ±0.21
<i>oes10</i>	<b>0.82 ± 0.05</b> 16 ± 5	0.80 ± 0.07 16	0.38 ± 0.07 32	0.59 ± 0.15 14 ± 7	0.43 ± 0.12 16	0.48 ±0.12	0.76 ±0.08
<i>edm</i>	<b>0.51 ± 0.18</b> 30 ± 11	0.21 ± 0.09 2	-0.17 ± 0.45	0.38 ± 0.19 16 ± 5	0.18 ± 0.26 2	0.35 ±0.19	0.26 ±0.21
<i>jura</i>	<b>0.62 ± 0.08</b> 21 ± 3	0.30 ± 0.10 3	0.57 ± 0.06 6	0.38 ± 0.11 23 ± 1	0.18 ± 0.15 3	0.60 ±0.05	0.61 ±0.06
<i>wq</i>	<b>0.14 ± 0.01</b> 76 ± 9	0.12 ± 0.01 14	-0.35 ± 0.08 28	0.09 ± 0.02 29 ± 1	-0.01 ± 0.01 14	0.08 ±0.02	0.13 ±0.03
<i>enb</i>	<b>0.99 ± 0.01</b> 118 ± 4	0.86 ± 0.02 2	0.91 ± 0.01 4	0.86 ± 0.01 13 ± 1	0.98 ± 0.01 2	<b>0.99</b> ±0.01	<b>0.99</b> ±0.08

In particular, we can see that KSSHIBA consistently outperforms most reference methods in every database, pointing out the performance advantages obtained in *edm* and *oes97*. Additionally, it is remarkable that this performance improvement is accomplished with an effective dimensional reduction, since KSSHIBA, applying a feature extraction, is able to outperform both a SVR and a MLP that use all the original features. At the same time, the results obtained by KSSHIBA with  $K_c = C$  reveal that standard KSSHIBA is being too conservative in the number of extracted features and we could force a more restrictive pruning without degrading the final performance (note that KSSHIBA with  $K_c = C$  only deteriorates in the problems with only 2 or 3 (*edm*, *jura* and *enb*) since in this cases the number of latents is extremely reduced).

### 3.3 Evaluation of the solution in terms of RVs

Now, we want to test the capabilities of the KSSHIBA approach to automatically construct compact solutions by selecting a subset of training points, in other words, using RVs. For this purpose, we use the same databases and experimental setup as Section 3.2, but we compare with KPCA+LR and KCCA+LR using a Nyström [33] subsampling technique. In this way, KPCA+LR and KCCA+LR models will use this subsampling to select RV subsets; in these cases, the optimum percentage of RVs (respect to the total number of training data) has been selected by CV exploring their values in the set  $[1, 2, 3, 4, 5, 10, \dots, 100]\%$ .

Table 5 shows that the inclusion of the automatic RV selection on KSSHIBA keeps the original model performance for most databases, even improving it for *oes97* and *edm*. This is done while the model complexity is drastically reduced; in fact, analysing these results in detail, it is observed that the fact of reducing the number of RVs favours an additional reduction in the final number of latent factors. When comparing to KPCA+LR and KCCA+LR, we can observe that KSSHIBA tends to show a lower percentage of RVs to describe the kernel. This is due to the fact that KSSHIBA learns the relevance of each element and eliminates them accordingly, whereas KPCA and KCCA obtain this compact solutions with a random selection of RVs.

Table 5: Results on the multitask databases for the automatic SV selection. The first subcolumn shows on the white subrow the mean and standard deviation of the R2 score and on the gray subrow the number of effective latent factors ( $K_c$ ), the second subcolumn includes the percentage of RVs selected( $\%RV_s$ ).

	Sparse KSSHIBA		KPCA + LR		KCCA + LR	
	R2 - $K_c$	$\%RV_s$	R2 - $K_c$	$\%RV_s$	R2 - $K_c$	$\%RV_s$
<i>at1pd</i>	$0.77 \pm 0.09$ 41 ± 11	18.4 ± 24.1	$0.78 \pm 0.09$ 87 ± 35	69.7 ± 32.9	<b><math>0.80 \pm 0.09</math></b> 6	84.8 ± 27.5
<i>at7pd</i>	$0.55 \pm 0.15$ 70 ± 27	18.5 ± 26.3	$0.56 \pm 0.18$ 90 ± 37	79.7 ± 31.7	<b><math>0.60 \pm 0.12</math></b> 6	73.9 ± 34.1
<i>oes97</i>	<b><math>0.58 \pm 0.15</math></b> 61 ± 7	38.6 ± 24.5	$0.52 \pm 0.24$ 124 ± 34	81.7 ± 27.8	$0.42 \pm 0.30$ 16	23.9 ± 27.8
<i>oes10</i>	<b><math>0.77 \pm 0.11</math></b> 74 ± 6	44.4 ± 38.4	$0.71 \pm 0.12$ 132 ± 53	71.9 ± 11.6	$0.66 \pm 0.10$ 16	57.8 ± 35.2
<i>edm</i>	<b><math>0.42 \pm 0.21</math></b> 13 ± 4	53.8 ± 28.5	$0.41 \pm 0.26$ 29 ± 14	52.5 ± 30.5	$0.20 \pm 0.14$ 2	22.7 ± 13.6
<i>jura</i>	<b><math>0.58 \pm 0.14</math></b> 30 ± 4	48.7 ± 38.4	$0.57 \pm 0.10$ 59 ± 14	60.7 ± 28.9	$0.36 \pm 0.09$ 3	18.9 ± 7.5
<i>wq</i>	<b><math>0.12 \pm 0.01</math></b> 21 ± 2	58.1 ± 33.2	$0.12 \pm 0.02$ 96 ± 49	22.9 ± 15.9	$0.10 \pm 0.01$ 14	5.9 ± 3.1
<i>enb</i>	<b><math>0.99 \pm 0.01</math></b> 78 ± 8	19.5 ± 12.8	$0.91 \pm 0.01$ 28 ± 1	48.9 ± 32.9	$0.97 \pm 0.01$ 2	41.9 ± 12.2

To complete this analysis, Figure 3 includes the mean R2 over 10 folds varying the percentage of RVs. In this case, we show the results with the two databases for which SSHIBA is outperformed, the results with the other databases are available in B. For the sake of comparison, we also included the MRD results when its percentage of inducing points is varied. On the one hand, we can observe that the MRD behavior w.r.t. to the number of inducing points is unstable and quite dependent on the database. Further, the position of inducing points uses a regular grid, since this implementation does not allow their optimization, causing high fluctuations in its performance. On the other hand, while KPCA+LR and KCCA+LR present fluctuations in their performance requiring to adjust the number of RVs to obtain an accurate performance, KSSHIBA has a relatively constant R2 value. This phenomenon occurs because KSSHIBA learns the relevance of each SV and weight their influence on the update of the parameters during all the model inference.

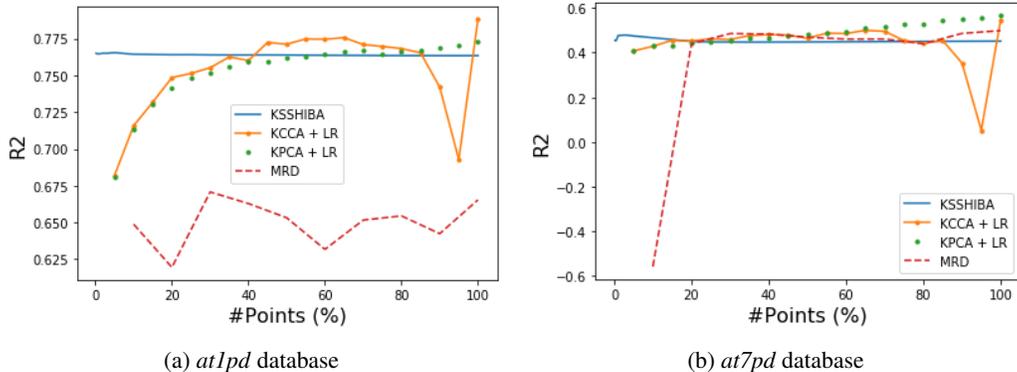


Figure 3: R2 results with different percentages of RVs in KSSHIBA, KCCA+LR and KPCA+LR or inducing points in MRD.

### 3.4 Analysis of the feature relevance

In order to test the feature relevance functionality (see Section 2.3), we now study KSSHIBA on classification images database, using the image as input view and the category label as the output view, as this scheme provides an illustrative mechanism to analyze the feature (pixel) relevances. In particular, we used an aligned version of the Labeled Faces in the Wild (LFW) dataset [34] obtained by [35] and the *warpAR10P* ( $60 \times 40$  pixels), *Yale* ( $32 \times 32$  pixels) and *Olivetti* ( $32 \times 32$  pixels) databases, which can be found in the Feature Selection Repository<sup>2</sup>. Whereas the later databases were preprocessed, in LFW we had to crop the images to eliminate undesirable information and resize them to  $60 \times 40$  pixels to reduce the computational cost of training the models; besides, to limit the size of the database we only used the images of the 7 people with most images in the database. The characteristics of these databases are described in Table 6.

Table 6: Characteristic of the faces databases used in this work.

Database	Samples	Features	Classes
<i>LFW</i>	1,277	2,400	7
<i>warpAR10P</i>	130	2,400	10
<i>Yale</i>	165	1,024	15
<i>Olivetti</i>	400	1,024	40

The KSSHIBA scheme followed in this extension consist in including an ARD kernel in the input images to obtain a feature relevance analysis. This way, Figure 4 shows the relevance mask learnt by the model, over echa dataset, having a lighter colour for relevant pixels a darker colour when the pixel is not relevant. With these databases we can easily recognise the learnt face shape, which determine which pixels the model have to center on, paying less attention to the background and, in some cases, clearly defining the nose, cheek or chin, i.e., defining a mask of the face.

Analysing these masks we can see that the relevances learnt for *LFW* is more informative and have a higher definition than the other, mainly because there is a considerably higher number of samples for this database. Both the *warpAR10P* and the *Olivetti* databases agree to focus on the area related to the glasses as there is representative number of images with glasses in both databases. Besides, we have that if one subject wears glasses in one image, he will also wear glasses in the rest, therefore having a relevant area to classify the different subjects in these datasets. Another fact that stands out for us is that the mouths and eyes are generally considered as non-relevant for the classification task, while mainly focusing on the hair, cheeks and the face shape. This is not so accentuated in the *warpAR10P* dataset because there are a considerable number of images with cloths covering the face under the nose.

<sup>2</sup><http://featureselection.asu.edu/datasets.php>

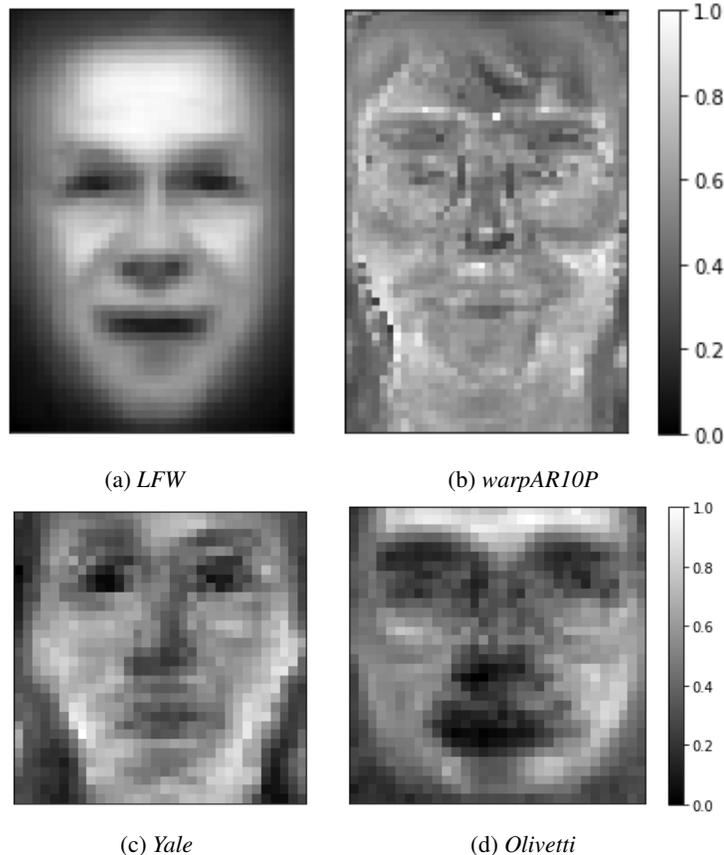


Figure 4: Feature masks learnt by the feature selection extension of KSSHIBA for different face recognition problems. The mask represent the importance of each pixel: lighter colours imply the pixel is more relevant while darker ones represent the pixel is less relevant.

### 3.5 Analysis of the extracted latent factors

In this section we want to evaluate the interpretability of the extracted latent factors obtained by the proposed model in comparison to the MRD approach based on shared GPLVMs. In particular, we will analyse both of them on the *Oil* classification database [36] (which has 2,000 samples, 12 features and 3 output classes). For this purpose we have trained both models with 15 latent factors (number of features plus output classes) combined with ARD latent factor selection. KSSHIBA uses a RBF kernel only for the input view meanwhile MRD uses it for both their input and output views. Under these conditions, the accuracy in the prediction of the labels for the MRD was of 99.0% and KSSHIBA achieved a 99.4%. Furthermore, with the available MRD implementation (*Matlab*), the computational time is not scalable for the number of data.

Figure 5 shows the learnt relevance for each latent factor for MRD ( $W$ ) and KSSHIBA ( $\gamma^{(m)}$ ). In the case of MRD, the formulation limits the model to specify the number of latent factors that are going to be used for the input and for the output data (Figure 5a shows the relevance for all these common factors). This means that the first 12 latent factors are related to the input view and the last 3 to the output, seeing that the model mainly focuses on the latter. On the other hand, KSSHIBA presents independent weights for each view (see Figures 5b and 5c). This implies that the latent factors might be not relevant and could be pruned (latent 7), are only relevant for one view (latents 5 and 14) or are simultaneously relevant for more than one view (highlighting latents 0, 2 and 8). This latent information obtained by the inclusion of an ARD prior over the projection matrices  $\mathbf{W}^{(m)}$ , provides a more interpretable model.

Finally, to analyse the interpretability of the results, Figure 6 shows the most relevant extracted latent features with both models. For this three classes classification problem, we can see that the KSSHIBA

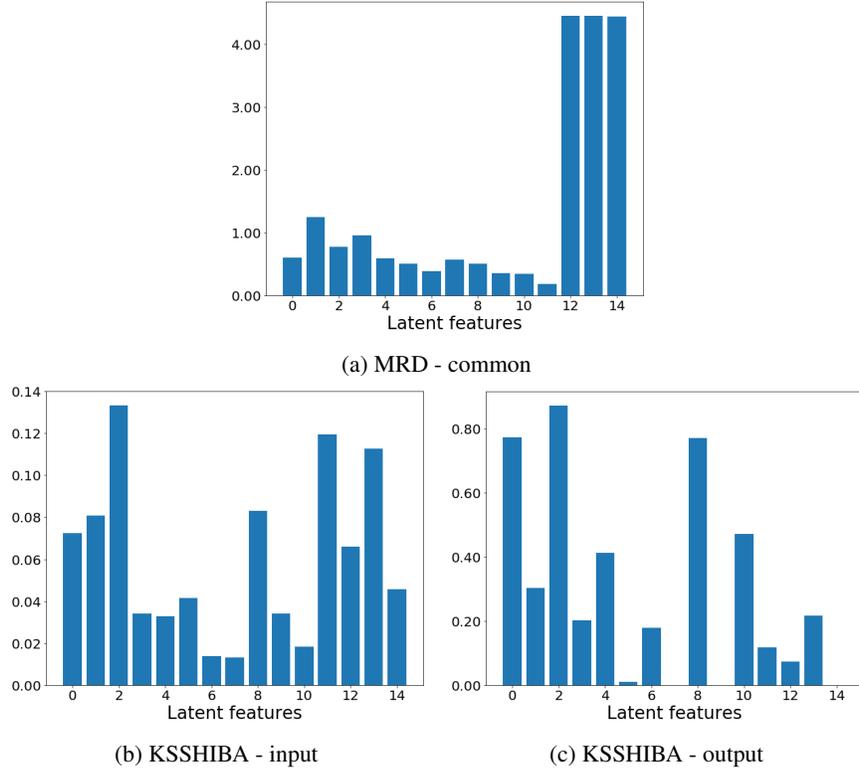


Figure 5: Measure of relevance for each learnt latent factor on the *Oil* database. Figure 5a shows the relevance of the commons for MRD model (all latents have resulted to be shared by both views). Figures 5b and 5c show, respectively, the relevance for the input view and the output view for KSSHIBA.

algorithm is capable of finding a subspace where, with the three most relevant common latent factors, the classification problem is easily solved. Meanwhile, MRD with three latent factors projects most samples into a single point, needing more latent factors to discriminate the different classes.

We think that MRD needs for a large number of common latent to obtain a discriminative space due to the inclusion of a non-linearity in its output view. In fact, this suggests that using a simple linear output view in KSSHIBA facilitates the achievement of not only a better performance, but also a more informative and discriminative latent space.

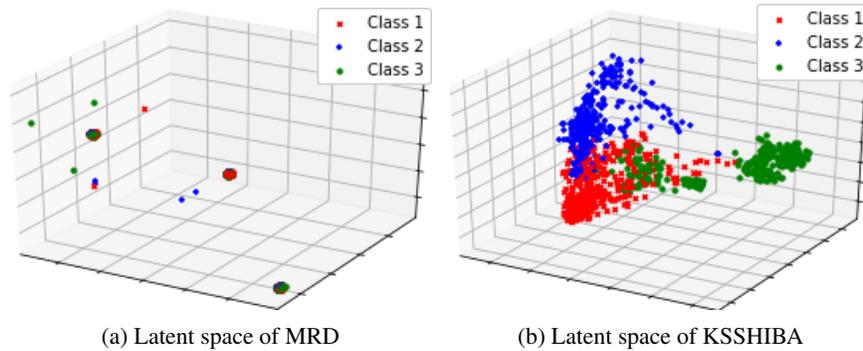


Figure 6: Learnt projections for the *Oil* database. Each figure shows the projections over the three most relevant factors: latents 12, 13 and 14 for MRD and latents 0, 2 and 8 for KSSHIBA.

### 3.6 Multiple kernel learning for factor analysis

One of the main functionalities of KSSHIBA is its capacity to combine multiple views into a single model. We can take advantage of this property when reconstructing kernel representations to combine different types of kernels (one per view) and obtain a latent representation resulting from the linear combination of different kernels, known as MKL. Furthermore, the nature of the algorithm allows us to automatically learn the relevance of each kernel while combining different kernels in one model.

To analyse this functionality, we decided to test it with three different databases: *Arrhythmia* and *Landsat* from the UCI repository [37] and *Fashion MNIST* [38]. These databases are described in Table 7.

Table 7: Characteristic of the databases used for MKL.

Database	Samples	Features		Classes
		Kernel 1	Kernel 2	
<i>Arrhythmia</i>	452	15	264	2
<i>Landsat</i>	6,435	4	4	6
<i>Fashion MNIST</i>	1,000	784	784	10

To analyze the MKL KSSHIBA performance, we have compare it with a SVM classifier and a CCA to extract the latent factors and then a SVM to classify (CCA+SVM), using as input a linear combination of two different kernels:

$$K(X_{i,:}, X_{j,:}) = \mu K^1(X_{i,S}, X_{j,S}) + (1 - \mu) K^2(X_{i,R}, X_{j,R}) \tag{14}$$

where  $K^1$  is the first kernel,  $K^2$  is the second kernel and  $\mu \in [0, 1]$  is the combination coefficient. Using these equations for the *Arrhythmia* database, we defined  $K^1$  as a RBF kernel with the demographics and general ECG information (15 features) and  $K^2$  as a RBF kernel with the channel parameters like width and amplitude (263 features). For the *Landsat* database we defined  $K^1$  as a RBF kernel with the spectral information (4 features) while  $K^2$  was a RBF kernel uses the contextual information (4 features), as proposed in [39]. Finally, for the *Fashion MNIST* database we used only 1,000 samples for the experiment, with 784 features, using the same data for each kernel, setting  $K^1$  to be RBF and  $K^2$  to be polynomial. We validated the  $C$  and  $\gamma$  parameters for the SVM as well as the combination coefficient  $\mu$ . For CCA we set the number of latent factors to  $\#classes - 1$ .

Table 8: Multiple Kernel Learning analysis. The performance measure used is the multiclass AUC, which we calculated in a 10-fold cross-validation.

Database	MKL SSHIBA	MKL SSHIBA + RVS	SVM	CCA+SVM
<i>Arrhythmia</i>	0.809 ± 0.057	<b>0.818 ± 0.046</b>	0.774 ± 0.045	0.737 ± 0.059
<i>Landsat</i>	<b>0.983 ± 0.002</b>	<b>0.983 ± 0.002</b>	0.963 ± 0.042	0.980 ± 0.002
<i>Fashion MNIST</i>	0.951 ± 0.011	0.952 ± 0.013	<b>0.970 ± 0.010</b>	0.844 ± 0.012

The performance obtained with these configurations are included in Table 8, where we used two versions of KSSHIBA, with and without RVs Selection (RVS), and multiclass AUC as the performance measure. The results prove that the combination of the different kernels carried out by MKL-SSHIBA is capable of providing good results with respect to the baselines. In particular, in the *Arrhythmia* dataset we obtained an improvement of up to 0.044 with respect to the SVM. Furthermore, we can see that the inclusion of the RV selection provides a more robust model, as it automatically selects the RVs that are relevant for the problem. Although in the *Fashion MNIST* database the SVM provides a better result than the proposed model, we consider that the improvement is not enough to justify the lack of feature extraction; in fact, the results obtained by SVM with feature extraction are considerably lower than the obtained by the proposed model.

To further analyse the capabilities of the presented model, we decided to use the *Fashion MNIST* database with three different kernels to analyse the learnt latent factors. In particular, we decided

to combine the RBF and polynomial with a linear kernel. We included the labels in another view and used them as an output view. The performance obtained with this framework is equivalent to the one previously explored, probing that the combination of different kernels do not deteriorate the performance of the model.

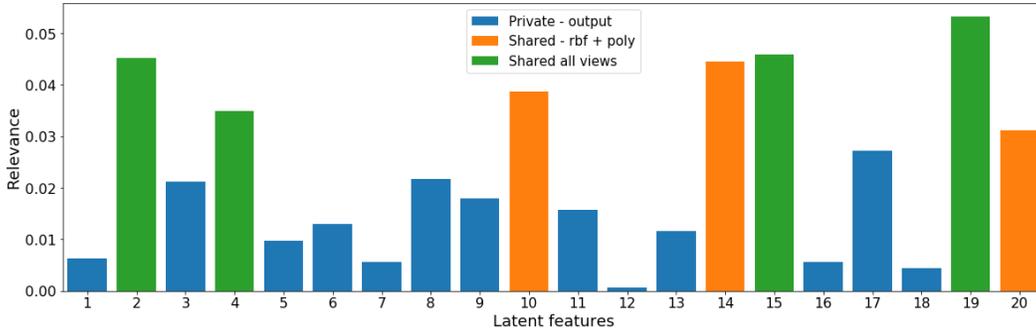


Figure 7: Relevance of the learnt latent factors. Measure of relevance on *Fashion MNIST* database combining kernels and labels on different views.

Figure 7 shows the relevance of each latent factor in the joint scenario (all kernels used). From the original 100 latent factors, the RBF and the polynomial kernels share 3 and the output view only uses 17, being most of them private. In fact, we can observe that the improved performance of the model is obtained using only four common factors to all views for the prediction of the output label. Note that, only the private output and the shared factors are used for the prediction, while the rest capture relations between views.

## 4 Conclusions

We propose a novel probabilistic latent variable model to generate kernel relationships, instead of data observations, based on a linear generative model. We introduce this model using the Bayesian Inter-Battery Factor Analysis (BIBFA) approach proposed in [15] to show its capabilities to efficiently face semi-supervised heterogeneous multi-view problems combining linear and non-linear data representations. Besides, we extend the model formulation to provide the automatic selection of RVs, obtaining scalable solutions, as well as include an ARD prior over the kernel to obtain the feature relevance functionality. The model performance is evaluated in multi-dimensional regression, feature relevance over images and multiple-kernel learning problems demonstrating that the inclusion of kernelized observations provide fruitful results.

The results prove the relevance of the proposed formulation, achieving a not only competitive performance, but also transforming the data to a reduced set of interpretable latent variables and a compact model consisting in a reduced subset of RVs. Furthermore, the feature relevance criteria is able to learn relevant masks which provide insight knowledge of the input space for the goal task. Finally, we also proved that taking advantage of the multiview nature of KSSHIBA we can easily combine different kernels to enhance, even more, the final performance of the model. This model contributes to the field of Artificial Intelligence by providing a model capable of adapting to a wide range of scenarios. In particular, the ability of combining feature selection, with heterogeneous data, imputation of missing values and combination of multiple kernels can be very useful for some biomedic or face recognition problems. In these, there usually are a considerable number of missing values, the interpretability given by the feature relevance and the learnt latent factors might improve the analysis of the results and MKL allows the model to combine linear and non-linear data.

## Acknowledgments

The work of Pablo M. Olmos is supported by Spanish government MINECO under grant RTI2018-099655-B-10, by Comunidad de Madrid under grants IND2017/TIC-7618, IND2018/TIC-9649, and Y2018/TCS-4705, by BBVA Foundation under the Deep-DARWiN project, and by the European Union (FEDER and the European Research Council (ERC) through the European Unions Horizon

2020 research and innovation program under Grant 714161). C. Sevilla-Salcedo and V. Gómez-Verdejo’s work has been partly funded by the Spanish MINECO grant TEC2017-83838-R. The work of Alejandro Guerrero is supported by MINECO and EU FEDER.

## References

- [1] J. C. Loehlin, Latent variable models. hillsdale, nj: erlbaum, 1987.
- [2] P. K. Atrey, M. A. Hossain, A. El Saddik, and M. S. Kankanhalli, “Multimodal fusion for multimedia analysis: a survey,” Multimedia systems, vol. 16, no. 6, pp. 345–379, 2010.
- [3] A. Sharma, A. Kumar, H. Daume, and D. W. Jacobs, “Generalized multiview analysis: A discriminative latent space,” in 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2160–2167, IEEE, 2012.
- [4] J. Li, B. Zhang, G. Lu, and D. Zhang, “Generative multi-view and multi-feature learning for classification,” Information Fusion, vol. 45, pp. 215–226, 2019.
- [5] K. Pearson, “Liii. on lines and planes of closest fit to systems of points in space,” The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, vol. 2, no. 11, pp. 559–572, 1901.
- [6] H. Hotelling, “Relations between two sets of variates,” Biometrika, vol. 28, pp. 321–377, 12 1936.
- [7] B. Schölkopf, A. Smola, and K.-R. Müller, “Nonlinear component analysis as a kernel eigenvalue problem,” Neural computation, vol. 10, no. 5, pp. 1299–1319, 1998.
- [8] Y. Zhang, J. Zhang, Z. Pan, and D. Zhang, “Multi-view dimensionality reduction via canonical random correlation analysis,” Frontiers of Computer Science, vol. 10, no. 5, pp. 856–869, 2016.
- [9] S. Yu, L.-C. Tranchevent, B. De Moor, and Y. Moreau, Kernel-based data fusion for machine learning. Springer, 2013.
- [10] H. H. Harman, Modern factor analysis. University of Chicago press, 1976.
- [11] M. E. Tipping and C. M. Bishop, “Probabilistic principal component analysis,” Journal of the Royal Statistical Society: Series B (Statistical Methodology), vol. 61, no. 3, pp. 611–622, 1999.
- [12] S. Yu, K. Yu, V. Tresp, H.-P. Kriegel, and M. Wu, “Supervised probabilistic principal component analysis,” in Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 464–473, 2006.
- [13] J. Bernardo, M. Bayarri, J. Berger, A. Dawid, D. Heckerman, A. Smith, and M. West, “Bayesian factor regression models in the “large p, small n” paradigm,” Bayesian statistics, vol. 7, pp. 733–742, 2003.
- [14] A. Klami, S. Virtanen, and S. Kaski, “Bayesian canonical correlation analysis,” Journal of Machine Learning Research, vol. 14, no. Apr, pp. 965–1003, 2013.
- [15] C. Sevilla-Salcedo, V. Gómez-Verdejo, and P. M. Olmos, “Sparse semi-supervised heterogeneous interbattery bayesian analysis,” arXiv preprint arXiv:2001.08975, 2020.
- [16] C. K. Williams and C. E. Rasmussen, Gaussian processes for machine learning, vol. 2. MIT press Cambridge, MA, 2006.
- [17] N. Lawrence, “Probabilistic non-linear principal component analysis with gaussian process latent variable models,” Journal of machine learning research, vol. 6, no. Nov, pp. 1783–1816, 2005.
- [18] A. C. Damianou, C. H. Ek, M. K. Titsias, and N. D. Lawrence, “Manifold relevance determination,” in Proceedings of the 29th International Conference on International Conference on Machine Learning, pp. 531–538, 2012.

- [19] A. Damianou, N. D. Lawrence, and C. H. Ek, “Multi-view learning as a nonparametric nonlinear inter-battery factor analysis,” arXiv preprint arXiv:1604.04939, 2016.
- [20] R. M. Neal, Bayesian learning for neural networks, vol. 118. Springer Science & Business Media, 2012.
- [21] M. Titsias, “Variational learning of inducing variables in sparse gaussian processes,” in Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics (D. van Dyk and M. Welling, eds.), vol. 5 of Proceedings of Machine Learning Research, (Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA), pp. 567–574, PMLR, 2009.
- [22] A. Wilson and H. Nickisch, “Kernel interpolation for scalable structured gaussian processes (kiss-gp),” in Proceedings of the 32nd International Conference on Machine Learning (F. Bach and D. Blei, eds.), vol. 37 of Proceedings of Machine Learning Research, (Lille, France), pp. 1775–1784, PMLR, 2015.
- [23] W. Lian, P. Rai, E. Salazar, and L. Carin, “Integrating features and similarities: Flexible models for heterogeneous multiview data.,” in AAAI, pp. 2757–2763, Citeseer, 2015.
- [24] M. E. Tipping, “Sparse bayesian learning and the relevance vector machine,” Journal of machine learning research, vol. 1, no. Jun, pp. 211–244, 2001.
- [25] K. P. Murphy, Machine learning: a probabilistic perspective. MIT press, 2012.
- [26] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, “Variational inference: A review for statisticians,” Journal of the American Statistical Association, vol. 112, no. 518, pp. 859–877, 2017.
- [27] I. M. de Diego, A. Muñoz, and J. M. Moguerza, “Methods for the combination of kernel matrices within a support vector framework,” Machine learning, vol. 78, no. 1-2, p. 137, 2010.
- [28] S. Qiu and T. Lane, “A framework for multiple kernel support vector regression and its applications to sirna efficacy prediction,” IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 6, no. 2, pp. 190–199, 2008.
- [29] G. Fung, M. Dundar, J. Bi, and B. Rao, “A fast iterative algorithm for fisher discriminant using heterogeneous kernels,” in Proceedings of the twenty-first international conference on Machine learning, p. 40, 2004.
- [30] E. Spyromitros-Xioufis, G. Tsoumakas, W. Groves, and I. Vlahavas, “Multi-target regression via input space expansion: treating targets as inputs,” Machine Learning, vol. 104, no. 1, pp. 55–98, 2016.
- [31] A. Karalič and I. Bratko, “First order regression,” Machine learning, vol. 26, no. 2-3, pp. 147–176, 1997.
- [32] S. Džeroski, D. Demšar, and J. Grbović, “Predicting chemical parameters of river water quality from bioindicator data,” Applied Intelligence, vol. 13, no. 1, pp. 7–17, 2000.
- [33] C. Williams and M. Seeger, “Using the nyström method to speed up kernel machines,” in Advances in Neural Information Processing Systems 13 (NIPS 2000) (T. Leen, T. Dietterich, and V. Tresp, eds.), pp. 682–688, MIT Press, 2001.
- [34] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, “Labeled faces in the wild: A database for studying face recognition in unconstrained environments,” Tech. Rep. 07-49, University of Massachusetts, Amherst, 2007.
- [35] L. Wolf, T. Hassner, and Y. Taigman, “Effective unconstrained face recognition by combining multiple descriptors and learned background statistics,” IEEE transactions on pattern analysis and machine intelligence, vol. 33, no. 10, pp. 1978–1990, 2010.
- [36] C. M. Bishop and G. D. James, “Analysis of multiphase flows using dual-energy gamma densitometry and neural networks,” Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, vol. 327, no. 2-3, pp. 580–593, 1993.

- [37] D. Dua and C. Graff, “UCI machine learning repository,” 2017.
- [38] H. Xiao, K. Rasul, and R. Vollgraf, “Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms,” 2017.
- [39] J. Amorós-López, L. Gómez-Chova, L. Guanter, L. Alonso, J. Moreno, and G. Camps-Valls, “Multitemporal fusion of landsat and meris images,” in 2011 6th International Workshop on the Analysis of Multi-temporal Remote Sensing Images (Multi-Temp), pp. 81–84, IEEE, 2011.

## A SSHIBA’s summary

Semi-supervised Sparse Heterogeneous Inter-battery Bayesian Analysis (SSHIBA) is based on Bayesian Inter-Battery Factor Analysis (BIBFA) presented in [14]. The main goal of both is to jointly project different data representations, defined as “views”, into a discriminative low-dimensional space. The joint probability density function of BIBFA can be defined as

$$\mathbf{z}_{n,:} \sim \mathcal{N}(0, I_{K_c}) \quad (15)$$

$$\mathbf{w}_{:,k}^{(m)} \sim \mathcal{N}\left(0, \left(\alpha_k^{(m)}\right)^{-1} I_{K_c}\right) \quad (16)$$

$$\mathbf{x}_{n,:}^{(m)} | \mathbf{z}_{n,:} \sim \mathcal{N}\left(\mathbf{z}_{n,:} \mathbf{W}^{(m)\top}, \tau^{(m)-1} I_{D_m}\right) \quad (17)$$

$$\alpha_k^{(m)} \sim \Gamma\left(a^{\alpha^{(m)}}, b^{\alpha^{(m)}}\right) \quad (18)$$

$$\tau^{(m)} \sim \Gamma\left(a^{\tau^{(m)}}, b^{\tau^{(m)}}\right) \quad (19)$$

The graphic model associated to both is included in Figure 8. However, SSHIBA presents certain updates to this formulation which allow the model to adapt to more scenarios [15]. In particular, it is able to combine semi-supervised learning with feature selection while being able to model categorical and multi-dimensional binary (multi-label) data, besides real data.

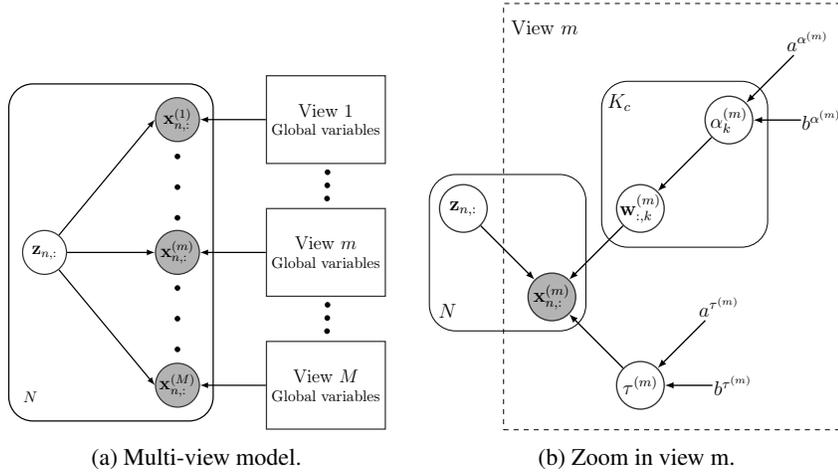


Figure 8: Plate diagram for the BIBFA and SSHIBA for real-data graphical model. Gray circles denote observed variables, white circles unobserved random variables. The nodes without a circle correspond to the hyperparameters.

## B Extended experiments

In this section we present a more extensive version of the results presented in the article. These results include some databases and baselines that we did not include due to the space limitations or the lack

of relevance of the results. In particular, Table 9 includes KCCA and a MLP where the second hidden layer works as a feature extractor, including a bottleneck of  $C$ .

Finally, Figure 9 presents the analysis of the effect of the number of RVs or inducing points (MRD) analysed in the evaluation of the solution in terms of RVs. In particular, we include here the results obtained for the databases not included in the main article. The results on MRD are not included for the  $wq$  database because the model iterations have not ended at the moment this material is done due to the high computational time required by the library.

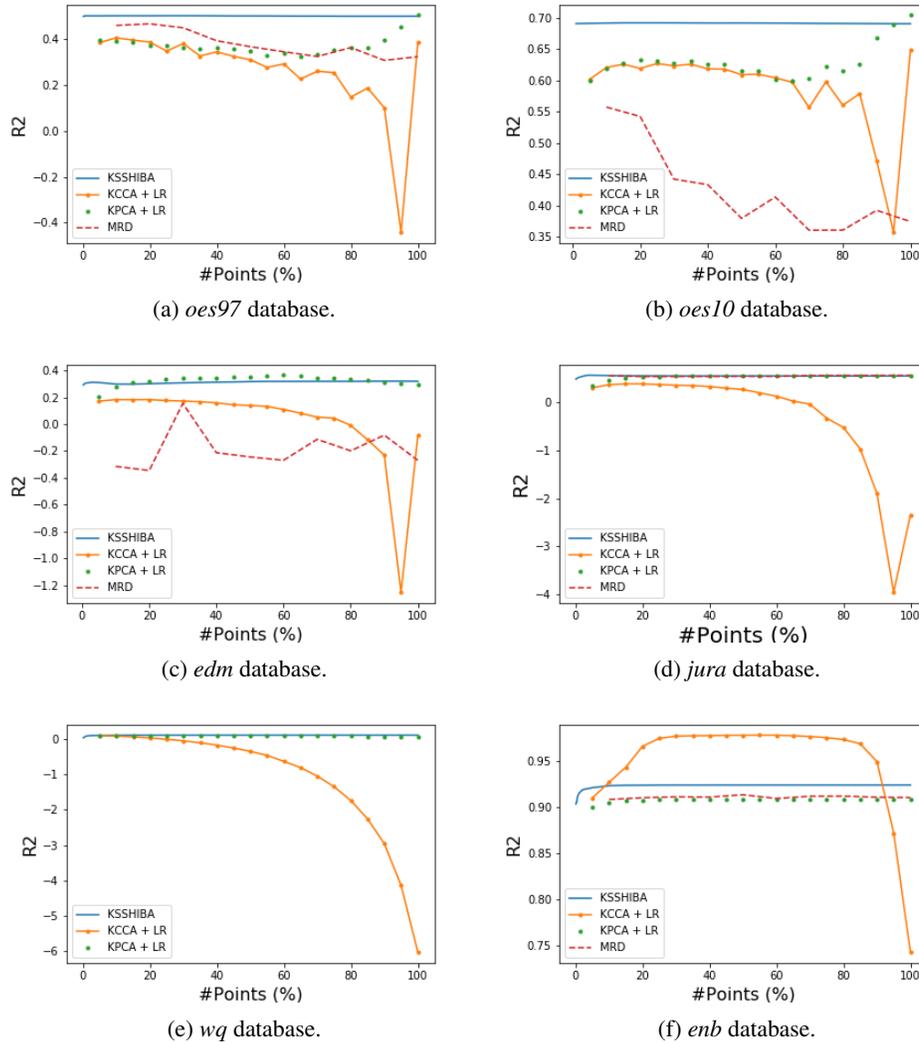


Figure 9:  $R^2$  results with different percentages of RVs in the the KSSHIBA, KCCA+LR and KPCA+LR or inducing points in MRD.

Table 9: Results on the multitask databases of the KSSHIBA and the different methods under study. In this case the data is normalised and the kernel, if applied, is centred. Each subrow represents the mean and standard deviation of the R2 score (white) and latent factor (light gray) used, respectively.

	KSSHIBA	KSSHIBA $K_c = C$	MRD	KPCA + LR	KCCA	KCCA + LR	SVR rbf	MLP $K_c = C$	MLP
<i>at1pd</i>	0.77 ± 0.09	<b>0.78 ± 0.09</b>	0.67 ± 0.07	0.67 ± 0.12	0.45 ± 0.05	0.75 ± 0.11	0.01 ± 0.05	0.75 ± 0.09	0.77 ± 0.12
	53 ± 8	6	12	22 ± 10	6	6		6	
<i>at7pd</i>	0.48 ± 0.26	0.52 ± 0.13	0.48 ± 0.12	0.39 ± 0.19	0.24 ± 0.05	<b>0.57 ± 0.16</b>	0.01 ± 0.03	0.29 ± 0.33	0.35 ± 0.69
	53 ± 11	6	12	21 ± 1	6	6		6	
<i>oes97</i>	0.63 ± 0.16	<b>0.69 ± 0.10</b>	0.34 ± 0.07	0.45 ± 0.20	0.30 ± 0.08	0.36 ± 0.09	0.39 ± 0.10	0.57 ± 0.22	0.58 ± 0.21
	108 ± 11	16	32	12 ± 7	16	16		16	
<i>oes10</i>	0.79 ± 0.08	<b>0.80 ± 0.07</b>	0.38 ± 0.07	0.59 ± 0.15	0.35 ± 0.17	0.43 ± 0.12	0.47 ± 0.12	0.77 ± 0.07	0.76 ± 0.08
	104 ± 22	16	32	14 ± 7	16	16		16	
<i>edm</i>	0.37 ± 0.19	0.21 ± 0.09	-0.17 ± 0.45	<b>0.38 ± 0.19</b>	0.26 ± 0.18	0.18 ± 0.26	0.35 ± 0.19	0.14 ± 0.17	0.26 ± 0.21
	17 ± 2	2	4	16 ± 5	2	2		2	
<i>jura</i>	<b>0.61 ± 0.10</b>	0.30 ± 0.10	0.57 ± 0.06	0.38 ± 0.11	0.11 ± 0.08	0.18 ± 0.15	0.60 ± 0.05	0.32 ± 0.12	<b>0.61 ± 0.06</b>
	64 ± 7	3	6	23 ± 1	3	3		3	
<i>wq</i>	0.12 ± 0.01	0.12 ± 0.01	-0.35 ± 0.08	0.09 ± 0.02	-0.01 ± 0.01	-0.01 ± 0.01	0.08 ± 0.02	0.10 ± 0.02	<b>0.13 ± 0.03</b>
	48 ± 3	14	28	29 ± 0.98	14	14		14	
<i>enb</i>	<b>0.99 ± 0.01</b>	0.86 ± 0.02	0.91 ± 0.01	0.86 ± 0.01	0.96 ± 0.01	0.98 ± 0.01	<b>0.99 ± 0.01</b>	0.89 ± 0.01	<b>0.99 ± 0.08</b>
	118 ± 4	2	4	13 ± 1	2	2		2	