REGRESSING KERNEL DICTIONARY LEARNING

Kriti Kumar, Angshul Majumdar† M Girish Chandra* and A Anil Kumar**

* TCS Research and Innovation, Bangalore, India.
† IIIT Delhi, New Delhi, India.
Email: {kriti.kumar, m.gchandra, achannaanil.kumar}@tcs.com, angshul@iiitd.ac.in

ABSTRACT

In this paper, we present a kernelized dictionary learning framework for carrying out regression to model signals having a complex non-linear nature. A joint optimization is carried out where the regression weights are learnt together with the dictionary and coefficients. Relevant formulation and dictionary building steps are provided. To demonstrate the effectiveness of the proposed technique, elaborate experimental results using different real-life datasets are presented. The results show that non-linear dictionary is more accurate for data modeling and provides significant improvement in estimation accuracy over the other popular traditional techniques especially when the data is highly non-linear.

Index Terms— Dictionary learning, kernel methods, method of optimal directions, regression

1. INTRODUCTION

We are living in the era of data deluge. Even though visual data has dominated/dominating this deluge, the equation is changing with huge data emanating from the Internet of Things (IoT)/ Machines. In order to understand the data and make effective use of them, it is necessary to have appropriate data-driven methods to capture the nature of data. With this understanding, one can carry out different inference tasks like, classification, clustering and regression.

Restricting to data modeling, many of the existing techniques from the data analysis community can be tried. For any data analysis, it is necessary to identify dependent variables also known as responses or predicands, and independent variables or predictors. The relationship between the predictors and responses is described by a regression function [1]. This function approximation approach is useful to model the data, to characterize different states of the data generating source. For example, for Computer Numerical Control (CNC) machines; given labeled data for normal operation, one can model the CNC machine performance (appropriate dependent variables) as a function of several other independent variables which can be from different sensors. The regression function learnt could then be used to asses the performance of CNC machine for an unknown test input. If the estimated and actual response is similar it depicts normal behavior else a change is detected. If the change is significant, it can be associated with abnormal/anomalous behavior with the help of additional information.

Since there is no "One fits All" solution for carrying out the modeling addressing different varieties of data, we need to have basket of techniques. Signal processing can provide systematic framework to arrive at new data-driven models. In this paper, we propose a kernel dictionary learning framework for carrying out regression.

Since its introduction, dictionary learning has been used profusely for analysis and synthesis problems especially arising in image processing [2], [3], [4], [5]. The basic formulation is given as:

$$\boldsymbol{X} = \boldsymbol{D}\boldsymbol{Z} \tag{1}$$

where, $\boldsymbol{X} \in \mathbb{R}^N$ is the data that is represented by the learnt dictionary or basis $\boldsymbol{D} \in \mathbb{R}^{N \times K}$ containing atoms as its columns and the learnt coefficients $\boldsymbol{Z} \in \mathbb{R}^K$.

The origin of dictionary learning lies in matrix factorization [6] and sparse coding [7]. The standard matrix factorization problem can be solved by the Method of Optimal Directions (MOD)[8] by alternately solving for the two variables:

$$\min_{\boldsymbol{D},\boldsymbol{Z}} \|\boldsymbol{X} - \boldsymbol{D}\boldsymbol{Z}\|_F^2 \tag{2}$$

For sparse coding problems, where $\boldsymbol{Z}$ is constrained to be sparse, K-SVD algorithm [9] is more popular. It solves for a dictionary using (3) such that the coefficients are sparse.

$$\min_{\boldsymbol{D},\boldsymbol{Z}} \|\boldsymbol{X} - \boldsymbol{D}\boldsymbol{Z}\|_F^2 \; s.t. \|\boldsymbol{Z}\|_0 \leq \boldsymbol{\tau} \tag{3}$$

where, $\|\boldsymbol{Z}\|_0$ is the usual l_0 sparsity measure which counts the number of non-zero elements in $\boldsymbol{Z}$. This optimization results in a sparse representation of data which is learnt using maximum $\boldsymbol{\tau}$ non-zero entries of $\boldsymbol{Z}$.

The unsupervised version of sparse coding has been used profusely for solving inverse problems like denoising, deblurring, inpainting, reconstruction [10], [11], [12] etc. Machine learning researchers have used dictionary learning for feature extraction. But instead of using the basic unsupervised formulation discriminative penalties are added to (3) for improved analysis [7]. A slightly dated treatise on this topic is available at [13].

In standard dictionary learning, a dense dictionary needs to be learnt from the data. There are two issues with this approach 1) using limited data leads to overfitting; and 2) large scale problems cannot be handled owing to explicit computations with the dictionary. To address both these issues, doubly sparse dictionary learning [14] has been proposed. The basic idea is to express the dictionary as an arbitrary sparse linear combination of fixed basis (e.g. wavelet, DCT Fourier etc.). The model is expressed as:

$$\boldsymbol{X} = \boldsymbol{\Phi}\boldsymbol{A}\boldsymbol{Z} \tag{4}$$

Here, $\boldsymbol{\Phi}$ is a combination of some pre-defined basis, $\boldsymbol{A}$ is the combining weights that picks up the appropriate basis from $\boldsymbol{\Phi}$ to form the dictionary. Here, both $\boldsymbol{A}$ and $\boldsymbol{Z}$ need to be learnt. This is framed as:

$$\min_{\boldsymbol{A},\boldsymbol{Z}} \|\boldsymbol{X} - \boldsymbol{\Phi}\boldsymbol{A}\boldsymbol{Z}\|_F^2 \;\; s.t. \; \|\boldsymbol{Z}\|_0 \leq \boldsymbol{\tau} \;\; and \;\; \|\boldsymbol{A}\|_0 \leq \boldsymbol{\rho} \tag{5}$$

The concept of kernel dictionary learning [15] is somewhat related to doubly sparse dictionary learning; both of them express the

dictionary as a linear combination of a fixed basis. In kernel dictionary learning, the fixed basis is a non-linear combination of the data. It is essentially a non-linear mapping from $\mathbb{R}^N$ to high dimensional feature space $\mathcal{F}$. So, instead of expressing the original data, as in standard sparse dictionary learning, kernel dictionary learning expresses the non-linear version of the data in terms of a dictionary formed by a linear combination of the non-linear version of the data. Mathematically, this is expressed as [15]:

$$\boldsymbol{\varphi(X) = \varphi(X)AZ} \tag{6}$$

where, $\boldsymbol{\varphi(X)}$ is the matrix obtained by transforming $\boldsymbol{X}$ to a high dimensional feature space $\mathcal{F}$. The learning algorithm is expressed as:

$$\min_{\boldsymbol{A,Z}} \|\boldsymbol{\varphi(X) - \varphi(X)AZ}\|_F^2 \;\; s.t. \; \|\boldsymbol{Z}\|_0 \leq \boldsymbol{\tau} \tag{7}$$

The optimization problem in (7) is solved using alternate minimization like in the case of dictionary learning.

Few attempts have been made that make use of kernel dictionary learning framework for image classification tasks [16], [17]. Results demonstrate that exploiting non-linear sparsity via learning dictionaries in a non-linear feature space provides superior performance compared to their linear counterparts and kernel PCA. Recently, some works have also reported the use of a unified objective function to jointly learn the dictionary and sparse linear classifier/regressor [18], [19]. In [18], a label consistent K-SVD algorithm is presented to learn a discriminative dictionary for sparse coding for object recognition tasks. The work in [19] presented a fast method for sparse regression in the presence of missing data. Both these methods had better performance over other sparse coding based techniques as the label information was utilized for learning via joint optimization.

Motivated by the method of joint optimization and the need to handle non-linearities in the data, in this paper, Kernel Dictionary Learning framework for Regression (KDLR) is proposed. To the best of our knowledge, there has been no prior study on kernel dictionary learning based regression where the regression formulation is learnt within the dictionary learning framework. This technique is shown to outperform the traditional Linear Regression (LR), Kernel Regression (KR), Least Absolute Shrinkage and Selection Operator (LASSO) and Dictionary Learning based techniques for regression (DLR) especially when the data (times series) exhibits certain complex non-linear evolution.

To elaborate on the proposed framework and demonstrate its applicability for regression analysis, the rest of the paper is organized as follows. Section 2 describes the proposed kernelized dictionary learning based approach for regression. Subsequently, Section 3 discusses the performance of the proposed algorithm using different datasets and Section 4 concludes the work.

2. KERNEL DICTIONARY LEARNING FOR REGRESSION

Given a multi-variate data of N samples, let $\boldsymbol{X} \in \mathbb{R}^{L \times N}$ represent the independent variables of feature vector length L and $\boldsymbol{y} \in \mathbb{R}^N$ represent the dependent variable. We propose to incorporate a ridge regression penalty into the kernel dictionary learning framework for carrying out a joint optimization where the dictionary atoms, coefficients and the regression weights are learnt together. Kernelization takes care of the non-linearities in the system and hence a simple linear regression formulation is sufficient after the transformation. Mathematically, the proposed formulation is given as:

$$\min_{\boldsymbol{A,Z,w}} \|\boldsymbol{\varphi(X) - \varphi(X)AZ}\|_F^2 + \lambda\|\boldsymbol{y - wZ}\|_2^2 + \mu\|\boldsymbol{w}\|_2^2 \tag{8}$$

where, $\boldsymbol{\varphi(X)} = [\boldsymbol{\varphi(x_1)}, ..., \boldsymbol{\varphi(x_N)}]$, $\boldsymbol{A} \in \mathbb{R}^{N \times K}$ is the atom representation dictionary, $\boldsymbol{Z} \in \mathbb{R}^{K \times N}$ are the coefficients and $\boldsymbol{w} \in \mathbb{R}^K$ are the regression weights. It is to be noted that in (8), the sparsity term is *not* included since the focus of this work is on regression, where we would be considering undercomplete dictionaries. The sparsity penalty was carried forth in some of the earlier works [18], [20] on classification, largely because KSVD was used as the workhorse algorithm for the associated dictionary learning.

Like any machine learning technique, the proposed technique has a training phase where, the dictionary atoms, coefficients and regression weights are learnt and a test phase where, the learnt dictionary and regression weights are used for estimating the response variable. These two phases are explained in detail below.

2.1. Training Phase

We follow the standard alternating minimization approach to solve (8). The sub-problems required to be solved are:

$$\boldsymbol{A} \leftarrow \min_{\boldsymbol{A}} \|\boldsymbol{\varphi(X) - \varphi(X)AZ}\|_F^2 \tag{9}$$

$$\boldsymbol{Z} \leftarrow \min_{\boldsymbol{Z}} \|\boldsymbol{\varphi(X) - \varphi(X)AZ}\|_F^2 + \lambda\|\boldsymbol{y - wZ}\|_2^2 \tag{10}$$

$$\boldsymbol{w} \leftarrow \min_{\boldsymbol{w}} \lambda\|\boldsymbol{y - wZ}\|_2^2 + \mu\|\boldsymbol{w}\|_2^2 \tag{11}$$

Solving for $\boldsymbol{A}$ using (9) results in the same update as given in [15]:

$$\boldsymbol{A = Z^T(ZZ^T)^{-1}} \tag{12}$$

The update for $\boldsymbol{Z}$ is obtained by taking the derivative of the expression in (10) and equating it to 0. After some simple mathematical manipulations, one arrives at the modified normal equations (one of the contributions of this work):

$$\boldsymbol{(A^T\mathcal{K}(X,X)A + \lambda w^T w)Z = A^T\mathcal{K}(X,X) + \lambda w^T y} \tag{13}$$

Here, $\boldsymbol{\mathcal{K}(X,X)} \in \mathbb{R}^{N \times N}$ is the kernel matrix of finite dimension whose elements are computed from:
$\boldsymbol{\kappa(x_i, x_j) = \varphi(x_i)^T\varphi(x_j)} \; \forall i, j = 1, ..., N$.
Equation (13) has an analytic solution, but for large volume of data, the kernel matrix is huge so it is not advisable to invert explicitly. One can use a few steps of Conjugate Gradient (CG) to solve (13) instead.

The update for the regression weights $\boldsymbol{w}$ is trivial since it is a simple least squares problem and is given as:

$$\boldsymbol{w(\lambda ZZ^T + \mu I) = \lambda y Z^T} \tag{14}$$

where, $\boldsymbol{I}$ is an all ones matrix of size $K \times K$. This concludes the training phase.

2.2. Test Phase

During testing, given a new test sample $\boldsymbol{x_{test}}$, we estimate the corresponding dependent variable or output $\boldsymbol{\hat{y}_{test}}$, by first computing the corresponding feature $\boldsymbol{z_{test}}$. The model is expressed as follows:

$$\boldsymbol{\varphi(x_{test}) = \varphi(X)Az_{test}} \tag{15}$$

Note that the dictionary does not change from the training phase; it is still defined by the linear combination of the non-linear version of training data. The solution for $\boldsymbol{z_{test}}$ is formulated as:

$$\min_{\boldsymbol{z_{test}}} \|\boldsymbol{\varphi(x_{test}) - \varphi(X)Az_{test}}\|_F^2 \tag{16}$$

Following the derivation as before, one finds that,

$$\boldsymbol{A}^T\boldsymbol{\mathcal{K}}(\boldsymbol{X},\boldsymbol{X})\boldsymbol{A}\boldsymbol{z}_{\boldsymbol{test}} = \boldsymbol{A}^T\boldsymbol{\mathcal{K}}(\boldsymbol{x}_{\boldsymbol{test}},\boldsymbol{X})^T \tag{17}$$

where, $\boldsymbol{\mathcal{K}}(\boldsymbol{x}_{\boldsymbol{test}},\boldsymbol{X}) = [\boldsymbol{\kappa}(\boldsymbol{x}_{\boldsymbol{test}},\boldsymbol{x}_1),...\boldsymbol{\kappa}(\boldsymbol{x}_{\boldsymbol{test}},\boldsymbol{x}_N)]$. One can either solve (17) explicitly by computing the inverse (for small problems) or solve it efficiently using CG. Once the feature $\boldsymbol{z}_{\boldsymbol{test}}$ is obtained, it is multiplied by the learnt regression weights to get the $\hat{\boldsymbol{y}}_{\boldsymbol{test}}$.

$$\hat{\boldsymbol{y}}_{\boldsymbol{test}} = \boldsymbol{w}\boldsymbol{z}_{\boldsymbol{test}} \tag{18}$$

The pseudo code of the KDLR algorithm is presented in Algorithm 1.

Algorithm 1 Kernel Dictionary Learning for Regression (KDLR)

Input: Set of training data, $\boldsymbol{X} = \boldsymbol{X}_{\boldsymbol{train}}$, $\boldsymbol{y} = \boldsymbol{y}_{\boldsymbol{train}}$, $\boldsymbol{K}$ (size of dictionary), parameters ($\boldsymbol{\lambda}$, $\boldsymbol{\mu}$) and kernel function $\boldsymbol{\kappa}$, test data $\boldsymbol{x}_{\boldsymbol{test}}$

Output: Learnt dictionary $\boldsymbol{A}$, weight vector $\boldsymbol{w}$, estimated output $\hat{\boldsymbol{y}}_{\boldsymbol{test}}$

Initialization: Set $\boldsymbol{Z_0}$ to random matrix with real numbers between 0 and 1 drawn from a uniform distribution, $\boldsymbol{w_0} = \boldsymbol{y}/\boldsymbol{Z}$ and $\boldsymbol{A_0} = \boldsymbol{O}$, iteration $i = 1$

procedure
loop: Repeat until convergence (or fixed number of iterations $Maxitr$)
  $\boldsymbol{A_i} \leftarrow \boldsymbol{Z}_{\boldsymbol{i-1}}^T(\boldsymbol{Z_{i-1}}\boldsymbol{Z}_{\boldsymbol{i-1}}^T)^{-1}$
  Normalize each column in $\boldsymbol{A_i}$ to a unit norm
  $\boldsymbol{Z_i} \leftarrow$ update using $\boldsymbol{A_i}$ & $\boldsymbol{w_{i-1}}$ using (13)
  $\boldsymbol{w_i} \leftarrow \boldsymbol{\lambda y Z}_{\boldsymbol{i}}^T(\boldsymbol{\lambda Z_i}\boldsymbol{Z}_{\boldsymbol{i}}^T + \boldsymbol{\mu I})^{-1}$
  $i \leftarrow i + 1$
  if $\|\boldsymbol{A_i} - \boldsymbol{A_{i-1}}\|_F < Tol$ or $i == Maxitr$ **then**
    $\boldsymbol{z}_{\boldsymbol{test}} \leftarrow (\boldsymbol{A}^T\boldsymbol{\mathcal{K}}(\boldsymbol{X},\boldsymbol{X})\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{\mathcal{K}}(\boldsymbol{x}_{\boldsymbol{test}},\boldsymbol{X})^T$
    $\hat{\boldsymbol{y}}_{\boldsymbol{test}} \leftarrow \boldsymbol{w}\boldsymbol{z}_{\boldsymbol{test}}$
    close;
  else go to *loop*

3. PERFORMANCE STUDY AND DISCUSSION

In this section, we demonstrate the performance of the proposed framework of kernel dictionary learning for regression tasks. Apart from synthetic data, three real-life datasets are considered. The estimation results of the proposed KDLR algorithm are presented along with those obtained from Linear Regression (LR), Kernel Regression (KR) [21], Least Absolute Shrinkage and Selection Operator (LASSO) and traditional Dictionary Learning (DLR) framework for comparative study. DLR method considered similar joint optimization mentioned in (8) worked out for non-kernelized data. For all the datasets, 90% of the data samples are randomly selected (using 5-fold cross validation) for training and the remaining are used for testing. Parameters $\boldsymbol{\lambda}$ & $\boldsymbol{\mu}$ of KDLR are carefully tuned through extensive search for each dataset. Gaussian kernel has been used for both KR and KDLR methods.

A. Synthetic Dataset: The non-linear data simulated in [22], [23] is considered for evaluation by taking 3 predictors and 1 response variable. The data comprising of 500 samples is l_2 normalized before applying different regression methods. The estimation results of the response variable using different methods are presented in Fig 1. Table 1 summarizes the estimation results in terms of Mean Squared Error (MSE) and Pearson's Correlation Coefficient (PCC) for all the methods. It can be seen that KDLR is able to estimate the peaks much better than other methods and has the least MSE and highest PCC.

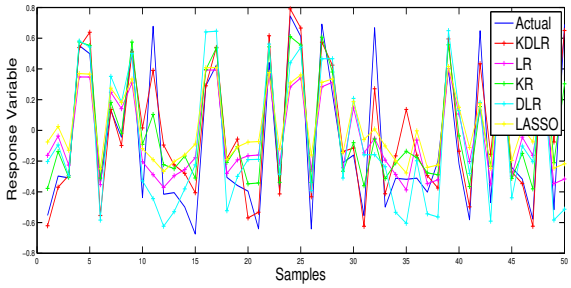

Fig. 1: Response variable estimation with different methods

Table 1: Results with Synthetic Dataset

Algorithm	MSE	PCC
KDLR ($K = 20, \sigma = 0.5$)	$\mathbf{0.036 \pm 0.010}$	**0.918**
KR	0.054 ± 0.013	0.873
LR	0.135 ± 0.019	0.607
DLR ($K = 3$)	0.134 ± 0.019	0.608
LASSO	0.137 ± 0.012	0.607

B. Public Datasets: Two UCI datasets are considered for regression analysis and are described in brief below.

(i) *Energy Efficiency* - This dataset is used to assess the heating load and cooling load requirements of buildings as a function of building parameters (Relative compactness, Surface area, Wall area, Roof area, Overall height, Orientation, Glazing area, Glazing area distribution) [24]. The dataset is comprised of 768 samples and 8 features as mentioned above from 12 different building shapes. Heating load is modeled as a function of the building parameters. The estimation results of the test data using different methods are presented in Fig 2. Table 2 gives the MSE and PCC of the estimation results obtained using different methods. Here again, KDLR is able to track the peaks in the heating load much better than its counterparts.

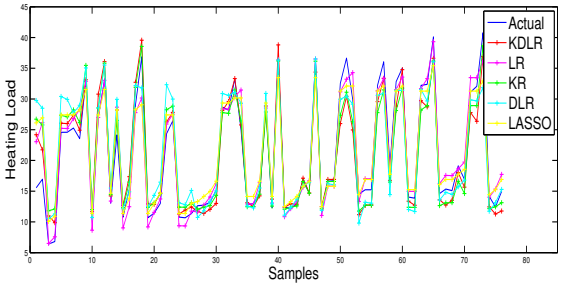

Fig. 2: Heating load estimation with different methods

(ii) *Wine Quality* - This dataset is used to model wine quality based on physicochemical tests [25]. It has 1599 samples of 11 physicochemical tests features for Red wine (Fixed acidity, Volatile acidity, Citric acid, Residual sugar, Chlorides, Free sulfur dioxide, Total sulfur dioxide, Density, pH, Sulphates, Alcohol). The data is normalized before applying it to different regression methods. Figure 3 presents the wine quality estimation results and Table 3 summarizes the estimation performance for different methods. The estimation results can also be visualized as a box plot, Fig. 4 gives a box plot of the MSE with the median value marked in red. It can be seen that the performance of proposed KDLR is closely similar to that of KR, but the median value of error for KDLR is low as compared to KR.

C. Smart Factory Dataset: This is a factory data acquired from Vertical Milling Center (VMC) which is a 4-axis CNC machine manufacturing precision machined components. In manufacturing,

Table 2: Results with Energy Dataset

Algorithm	MSE	PCC
KDLR ($K = 20, \sigma = 6$)	**7.985 ± 1.123**	**0.962**
KR	9.164 ± 1.652	0.955
LR	9.617 ± 2.407	0.953
DLR ($K = 5$)	19.133 ± 4.310	0.905
LASSO	13.382 ± 3.807	0.945

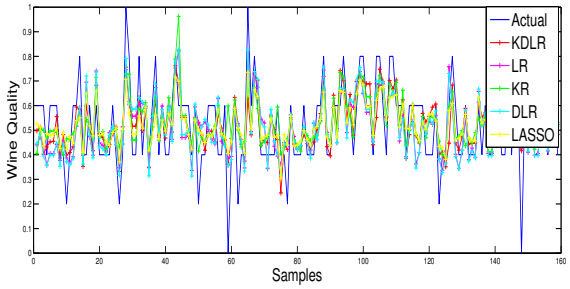

Fig. 3: Wine quality estimation with different methods

a component undergoes a number of machining operations like, roughing, milling, etc before resulting in a finished product. All these operations are performed by CNC machines. The performance of the machine can be studied by modeling the power and current (in terms of Spindle load and Axis-servoloads) consumed by the machine while performing a certain operation using a set defined machining parameters (Spindle speed, 3-axis Tool positions, 3-axis Feeds and Feed rate). This performance metric can be used to assess the health of the machine and condition of the cutting tools.

This dataset is comprised of 1 second sampled values of power, current and machining parameters used by the machine while performing a particular operation. Due to the massive size of the data, in our study 5817 samples are used for regression analysis which contains sufficient instances of similar machining operation for modeling. The kernel dictionary in KDLR, is learnt using $K = 10$ atoms and Gaussian kernel with σ = 2000. Traditional dictionary in DLR, is learnt using $K = 8$ atoms for carrying out regression. Figure 5 gives the comparative Spindle load estimation results for Pocket Rough operation using different regression methods. The box plot of the MSE in Fig. 6 demonstrate the superior performance of KDLR for modeling the machining operation over the linear methods. Its performance is however, closely similar to that KR but with less variance in the estimation error. This model has been used to ascertain abnormal operation of the machines in the real scenario, although the relevant results are not captured here.

Couple of remarks are worth noting. For any multi-variate analysis, the variables are appropriately normalized before subjecting them to processing. As is evident from the results, for the given normalization, the proposed technique performed better than its

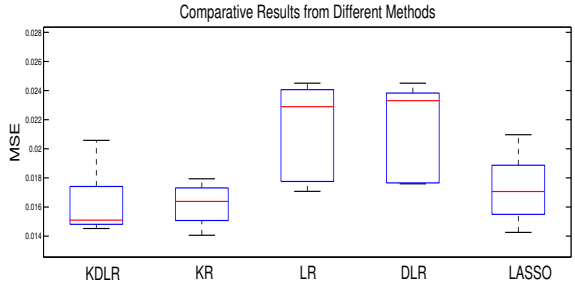

Fig. 4: Box plot of MSE for wine quality estimate

Table 3: Results with Wine Quality Dataset

Algorithm	MSE	PCC
KDLR ($K = 10, \sigma = 1.4$)	**0.016 ± 0.002**	**0.617**
KR	0.016 ± 0.001	0.641
LR	0.021 ± 0.003	0.487
DLR ($K = 10$)	0.021 ± 0.003	0.500
LASSO	0.017 ± 0.002	0.591

counterparts for the datasets considered (due to space limitation, the box plots are not provided for all the datasets). Secondly, the performance of dictionary-based regression depends on the number of atoms. For KDLR, with the first stage of non-linear transformation, larger dictionary (than the feature vector length) facilitates the accurate representation of the transformed features. Thus, one can expect good data modeling with an appropriately large dictionary, beyond which there will be marginal improvement (needless to say, it is data dependent). This fact is extensively studied for KDLR and DLR, and in comparing the performance, the appropriate size is accounted for each of them.

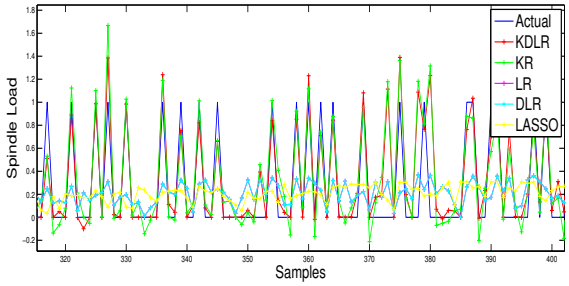

Fig. 5: Spindle load estimation with different methods

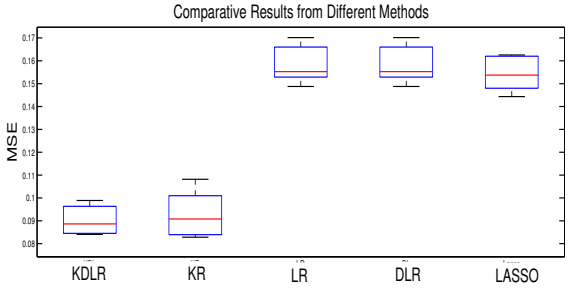

Fig. 6: Box plot of MSE for spindle load estimate

4. CONCLUSION

We have presented a kernelized dictionary learning approach for carrying out regression to model signals/time-series having complicated non-linear evolution. The relevant optimization formulation and the dictionary building steps are elaborated. Experimental results obtained with different real-life datasets demonstrate the potential of the proposed algorithm in effective modeling of the data. This technique offers significant improvement in estimation accuracy over the other popular traditional techniques. The work can be extended to handle multiple response variables. Also, one can consider deep dictionaries for more accurate modeling to represent the data. Additionally, one can also think of working out kernelized regressors using graph signal based dictionaries to effectively capture the complex inter-relationships among the multi-variate data samples.

5. REFERENCES

[1] Wolfgang Hardle, *Applied Nonparametric Regression*, Cambridge University Press, 1990.

[2] I. Tosic and P. Frossard, "Dictionary learning," *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 27–38, March 2011.

[3] Chenglong Bao, Hui Ji, Yuhui Quan, Zuowei Shen, undefined, undefined, undefined, and undefined, "Dictionary learning for sparse coding: Algorithms and convergence analysis," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 38, no. 7, pp. 1356–1369, 2016.

[4] Julien Mairal, Jean Ponce, Guillermo Sapiro, Andrew Zisserman, and Francis R. Bach, "Supervised dictionary learning," in *Advances in Neural Information Processing Systems 21*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds., pp. 1033–1040. Curran Associates, Inc., 2009.

[5] G. Chen and D. Needell, "Compressed sensing and dictionary learning," *Finite Frame Theory: A Complete Introduction to Overcompleteness*, vol. 73, 2016.

[6] Daniel D. Lee and H. Sebastian Seung, "Learning the parts of objects by nonnegative matrix factorization," *Nature*, vol. 401, pp. 788–791, 1999.

[7] Bruno A. Olshausen and David J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by v1?," *Vision Research*, vol. 37, no. 23, pp. 3311 – 3325, 1997.

[8] K. Engan, S. O. Aase, and J. Hakon Husoy, "Method of optimal directions for frame design," in *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No.99CH36258)*, 1999, vol. 5, pp. 2443–2446 vol.5.

[9] Michal Aharon, Michael Elad, and Alfred M. Bruckstein, "K-svd and its non-negative variant for dictionary design," in *Proceedings of the SPIE conference wavelets*, 2005, pp. 327–339.

[10] Yann LeCun, "Machine Learning and Pattern Recognition: Unsupervised Learning Sparse Coding," `https://www.cs.nyu.edu/~yann/2010f-G22-2565-001/diglib/lecture12-sparse-coding.pdf/`, 2010.

[11] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro, "Online dictionary learning for sparse coding," in *Proceedings of the 26th Annual International Conference on Machine Learning*, New York, NY, USA, 2009, ICML '09, pp. 689–696, ACM.

[12] J. Mairal, M. Elad, and G. Sapiro, "Sparse representation for color image restoration," *Trans. Img. Proc.*, vol. 17, no. 1, pp. 53–69, Jan. 2008.

[13] Ivana Tosic and Pascal Frossard, "Dictionary learning: What is the right representation for my signal?," *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 27–38, 2011.

[14] Ron Rubinstein, Michael Zibulevsky, and Michael Elad, "Double sparsity: learning sparse dictionaries for sparse signal approximation," *IEEE Trans. Signal Processing*, vol. 58, no. 3, pp. 1553–1564, 2010.

[15] H. V. Nguyen, V. M. Patel, N. M. Nasrabadi, and R. Chellappa, "Kernel dictionary learning," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2012, pp. 2021–2024.

[16] H. Van Nguyen, V. M. Patel, N. M. Nasrabadi, and R. Chellappa, "Design of non-linear kernel dictionaries for object recognition," *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 5123–5135, Dec 2013.

[17] A. Golts and M. Elad, "Linearized kernel dictionary learning," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 4, pp. 726–739, June 2016.

[18] Z. Jiang, Z. Lin, and L. S. Davis, "Label consistent k-svd: Learning a discriminative dictionary for recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 11, pp. 2651–2664, Nov 2013.

[19] R. Ganti and R. M. Willett, "Sparse Linear Regression With Missing Data," *ArXiv e-prints*, Mar. 2015.

[20] Ziyu Wang, Jianxiong Liu, and Jing-Hao Xue, "Joint sparse model-based discriminative k-svd for hyperspectral image classification," *Signal Processing*, vol. 133, no. Supplement C, pp. 144 – 155, 2017.

[21] Yi Cao, "Multivariant Kernel Regression and Smoothing," `https://in.mathworks.com/matlabcentral/fileexchange/19279`, March 2008.

[22] W. W. Hsieh, "Nonlinear canonical correlation analysis by neural networks," *Neural Netw.*, vol. 13, no. 10, pp. 1095–1105, Dec. 2000.

[23] William W. Hsieh, *Machine Learning Methods in the Environmental Sciences: Neural Networks and Kernels*, Cambridge University Press, New York, NY, USA, 1st edition, 2009.

[24] A. Xifara A. Tsanas, "Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools," in *Energy and Buildings*, 2012, vol. 49, pp. 560–567.

[25] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis., "Modeling wine preferences by data mining from physicochemical properties," in *In Decision Support Systems, Elsevier*, 2009, vol. 47(4), pp. 547–553.