Multi-task additive models with shared transfer functions based on dictionary learning

Alhussein Fawzi, Mathieu Sinn, and Pascal Frossard

***Abstract*—Additive models form a widely popular class of regression models which represent the relation between covariates and response variables as the sum of low-dimensional *transfer functions*. Besides flexibility and accuracy, a key benefit of these models is their *interpretability*: the transfer functions provide visual means for inspecting the models and identifying domain-specific relations between inputs and outputs. However, in large-scale problems involving the prediction of many related tasks, learning independently additive models results in a loss of model interpretability, and can cause overfitting when training data is scarce. We introduce a novel multi-task learning approach which provides a corpus of accurate and interpretable additive models for a large number of related forecasting tasks. Our key idea is to *share transfer functions across models* in order to reduce the model complexity and ease the exploration of the corpus. We establish a connection with sparse dictionary learning and propose a new efficient fitting algorithm which alternates between sparse coding and transfer function updates. The former step is solved via an extension of Orthogonal Matching Pursuit, whose properties are analyzed using a novel recovery condition which extends existing results in the literature. The latter step is addressed using a traditional dictionary update rule. Experiments on real-world data demonstrate that our approach compares favorably to baseline methods while yielding an interpretable corpus of models, revealing structure among the individual tasks and being more robust when training data is scarce. Our framework therefore extends the well-known benefits of additive models to common regression settings possibly involving thousands of tasks.**

***Index Terms*—Additive models, nonparametric regression, dictionary learning, sparse representations, multi-task learning.**

I. Introduction

Additive models are a widely popular class of nonparametric regression models which have been extensively studied theoretically and successfully applied to a wide range of practical problems in signal processing and machine learning [1], [2], [3]. The key ingredient of additive models are *transfer functions* that explain the effect of covariates on the response variable in an additive manner. Besides being flexible (e.g., allowing for the modeling of nonlinear effects for both continuous and categorical covariates) and yielding good predictive performance, an important selling point of additive models is their interpretability. In particular, the transfer functions provide intuitive visual means for application experts to understand the models and explore the relationship between input and output signals of the system under study.

In many real-world data modeling settings, one faces the problem of forecasting a large number (e.g., several thousands) of related tasks. In this case, learning additive models independently for each task has several disadvantages. Firstly, the number of models would be too large for a domain expert to visually inspect all the transfer functions, hence - in essence - the corpus of models loses its interpretability from a human point of view. Secondly, independently learning the models ignores *structure* and *commonality* among the tasks. Thirdly, when training data is scarce, learning the models independently is prone to overfitting the data.

To overcome these challenges, we introduce a novel *multi-task learning* framework for additive models. Intuitively, the key idea is to *share transfer functions across tasks* that exhibit commonality in their relationships between input and output variables. More specifically, each individual task is modeled as a weighted sum of transfer functions chosen from a candidate set which is common to all tasks, and the cardinality of which is small relative to the total number of tasks. Our algorithm for solving the multi-task additive model learning problem uses an intrinsic connection with *sparse dictionary learning* [4], [5], [6]. More specifically, we reformulate the fitting problem as a special form of dictionary learning with additional constraints; leveraging recent advances in the field, we propose a novel fitting approach that alternates between updates of the transfer functions and the weights that scale these functions. We introduce a novel algorithm for updating the coefficients that scale the transfer functions, called Block Constrained Orthogonal Matching Pursuit (BC-OMP), which extends conventional Orthogonal Matching Pursuit [7], [8]. Furthermore, we derive novel coherence conditions for the accurate recovery of the optimal solution which are interesting in their own right as they extend existing theory. Specifically, we show that under some conditions on the incoherence of the transfer functions, the proposed BC-OMP algorithm correctly identifies and selects the proper transfer functions. Transfer functions, which correspond to dictionary elements in our dictionary learning analogy, are updated using a traditional dictionary step update.

In the experimental part of our paper, we apply the proposed algorithm to synthetic and real-world electricity demand data. Synthetic results show that the proposed approach accurately learns transfer functions from noisy data. In addition, the proposed method is shown to outperform baseline linear and non-linear regression methods in terms of prediction accuracy. In a second experiment, we use a dataset of 4,066 smart meter time series data from Ireland, and show that our approach yields predictive performance comparable to baseline methods while only using a small number of candidate functions; interestingly, the discovered commonality of tasks corresponds to classes of residential and different types of enterprise

A. Fawzi and P. Frossard are with Ecole Polytechnique Federale de Lausanne (EPFL), Signal Processing Laboratory (LTS4), Lausanne, Switzerland (e-mail: alhussein.fawzi@epfl.ch; pascal.frossard@epfl.ch).

M. Sinn is with IBM Research, Dublin, Ireland.

customers. When using only a small fraction of the training data, our approach yields more robust results than independent learning and hence inherits the benefits of traditional multi-task learning. In a final experiment, we apply our multi-task learning algorithm to a *single-task* problem to improve the prediction accuracy of traditional additive model learning, while maintaining the number of learned transfer functions small.

Over the past decade, many works have shown the benefits of multi-task learning over independently learning the tasks [9], [10], [11], and different approaches to multi-task learning were considered. In [12], the authors impose the linear weight vectors of different tasks to be close to each other. The work in [13] constrains the weight vectors to live in a low-dimensional subspace. Still in the context of linear models, the authors of [14] assume that the tasks are clustered into groups, and that tasks within a group have similar weight vectors. In the context of additive models, [15] proposes new families of nonparametric models that enforce the selected covariates to be the same across tasks. This setting is particularly relevant for regression tasks involving a large number of covariates p, and the algorithm in [15] extracts a common set of covariates for the tasks. Our work significantly differs from [15] in several aspects. While [15] enforces a common set of covariates across tasks, the transfer functions are different. In other words, their approach only leverages commonality with respect to *which* covariates affect the dependent variable, but not *how* they affect it, leading to a number of transfer functions that is still too large for inspection by human experts. By imposing a common set of candidate transfer functions across tasks, we limit the number of transfer functions, and obtain interpretable models even for problems involving thousands of tasks. Moreover, unlike [15], we consider a setting where all covariates are relevant for the task at hand. Hence, in this paper, we are typically interested in problems involving a small number of input covariates p, and a large number of tasks N. Our approach shows that, by learning a number of transfer functions that is much smaller than pN, it is possible to achieve comparable or better performance than models involving a much larger number of parameters.

Finally, our work is also broadly related to dictionary learning methods for solving machine learning tasks. Methods for *discriminative* dictionary learning have previously been proposed in [16], [17], [18], [19], where the goal is to learn dictionaries that are specifically adapted to the classification task of interest. While we also use dictionary learning based techniques, our goal in this paper is significantly different: we consider the problem of extending the framework of additive models to multi-task settings, where transfer functions are shared. Dictionary learning and sparse coding have also been previously used in the context of multi-task learning. In [20], an online dictionary learning based technique is used to perform multi-task learning, where the assumed model is based on that of [21]. In particular, it is assumed that a (potentially nonlinear) parametric function relates the input covariates to the response variables, and that task-specific parameters are sparse linear combinations of a dictionary containing a library of latent components. To be amenable to efficient optimization, the model is simplified by taking the Taylor approximation of the parametric function, and SVD based dictionary learning techniques are further used [4]. In contrast to [20], the work we propose here focuses on a particular but important class of regression models, that is *additive models*. The structure of the regression problem lends to efficient solutions without requiring further approximations of the nonlinear model. Moreover, the sparsity model we use in this paper is very different from that of [20], as the proposed model is specifically adapted for the regression model of interest. We show the benefits of the proposed regression model on a series of toy and real world examples.

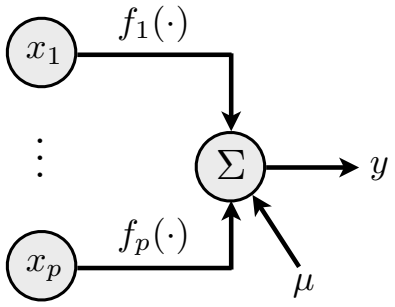


Fig. 1. Additive model diagram

The paper is structured as follows: Sec. II introduces notation and provides a review of additive models. In Sec. III we formulate the multi-task additive model learning problem and establish the connection with sparse dictionary learning. The algorithm for solving the multi-task problem is explained in Sec. IV. We provide in Sec. V novel recovery conditions guaranteeing the success of the proposed procedure for weights update, which is an important building block of the overall algorithm. In Sec. VI we describe our experiments on synthetic and real-world data; conclusions and an outlook on future research are given in Sec. VII.

II. Preliminaries

A. Notations

We use **boldface** notation for vectors and matrices. Moreover, we use $[n]$ to refer to the set $\{1, \ldots, n\}$. Given a vector $\mathbf{z}$, we denote by $\|\mathbf{z}\|_0$ the ℓ_0 "norm", that counts the number of nonzero elements in $\mathbf{z}$. Also, we denote by $\otimes$ the Kronecker product operation. If $\mathbf{Z} \in \mathbb{R}^{n_1 \times n_2}$ is a matrix, $\text{vec}(\mathbf{Z}) \in \mathbb{R}^{n_1 n_2}$ denotes the vectorization of $\mathbf{Z}$, obtained by stacking the columns of $\mathbf{Z}$, and $\mathbf{Z}^\dagger$ denotes the Moore-Penrose pseudo-inverse. Moreover, we use the notation $\mathbf{z} \in \mathbf{Z}$ to denote that $\mathbf{z}$ is one of the columns of $\mathbf{Z}$. Finally, $\mathbf{Z} \geq \mathbf{0}$ denotes the entry-wise non-negativity constraint.

B. Additive models review

We first briefly review additive models. Let $\{x_{ij}, i \in [n], j \in [p]\}$ and $\{y_i, i \in [n]\}$ denote respectively the observed covariates and response variable. Here, n is the number of

observations and p the number of covariates. Additive models have the form:

$$y_i = \mu + \sum_{j=1}^{p} f_j(x_{ij}) + \epsilon_i,$$

where μ is the intercept and ϵ_i is assumed to be a white noise process. The *transfer functions* f_j represent the effect of a covariate on the response variable. The additive model is illustrated in Fig. 1. To ensure unique identification of the f_j's, we assume that transfer functions are centered [22]: $\sum_{i=1}^{n} f_j(x_{ij}) = 0$ for all $j \in [p]$. Nonlinear transfer functions of continuous covariates are commonly modeled as smoothing splines [2], [1], i.e.,

$$f_j(z) = \sum_{t=1}^{T_j} \beta_{jt}\phi_{jt}(z), \tag{1}$$

where β_{jt} denotes the spline coefficients, ϕ_{jt} the B-spline basis functions, and T_j the number of basis splines. Using this representation, estimating the transfer functions therefore amounts to the estimation of the spline coefficients β_{jt} and the intercept μ. We consider the following fitting problem with centering constraints:

$$\min_{\mu,\{\beta_{jt}\}} \sum_{i=1}^{n} \left(y_i - \mu - \sum_{j=1}^{p}\sum_{t=1}^{T_j} \beta_{jt}\phi_{jt}(x_{ij}) \right)^2$$
$$\text{subject to } \sum_{i=1}^{n}\sum_{t=1}^{T_j} \beta_{jt}\phi_{jt}(x_{ij}) = 0 \text{ for all } j \in [p].$$

One can convert the above problem to an unconstrained optimization problem by centering the response and basis functions, as performed e.g. in [22].[1] Specifically, let $\bar{\phi}_{jt} = \frac{1}{n}\sum_{i=1}^{n} \phi_{jt}(x_{ij})$, $\mathbf{s}_j(z) = [\phi_{j1}(z) - \bar{\phi}_{j1}, \ldots, \phi_{jT_j}(z) - \bar{\phi}_{jT_j}]^T$ and

$$\mathbf{S} = \begin{bmatrix} \mathbf{s}_1(x_{11})^T & \ldots & \mathbf{s}_p(x_{1p})^T \\ & \vdots & \\ \mathbf{s}_1(x_{n1})^T & \ldots & \mathbf{s}_p(x_{np})^T \end{bmatrix}. \tag{2}$$

We define the vectorized spline coefficients $\boldsymbol{\beta} = \begin{bmatrix}\boldsymbol{\beta}_1^T & \ldots & \boldsymbol{\beta}_p^T\end{bmatrix}^T$ with $\boldsymbol{\beta}_j = \begin{bmatrix}\beta_{j1} & \ldots & \beta_{jT_j}\end{bmatrix}^T$. The above constrained fitting problem is then equivalent to the following unconstrained least squares problem:

$$\min_{\boldsymbol{\beta}} \|\mathbf{y} - \mathbf{S}\boldsymbol{\beta}\|_2^2,$$

where $\mathbf{y}$ denotes the centered response variables $\mathbf{y} = [y_1 - \bar{y}, \ldots, y_n - \bar{y}]^T$, with $\bar{y} = 1/n\sum_{i=1}^{n} y_i$. In order to avoid overfitting, a quadratic penalizer is commonly added, leading to the problem:

$$\min_{\boldsymbol{\beta}} \|\mathbf{y} - \mathbf{S}\boldsymbol{\beta}\|_2^2 + \boldsymbol{\beta}^T\boldsymbol{\Sigma}\boldsymbol{\beta},$$

with a regularization matrix $\boldsymbol{\Sigma}$. The penalized minimization problem has the closed form solution:

$$\hat{\boldsymbol{\beta}} = (\mathbf{S}^T\mathbf{S} + \boldsymbol{\Sigma})^{-1}\mathbf{S}^T\mathbf{y},$$

provided that $\mathbf{S}^T\mathbf{S} + \boldsymbol{\Sigma}$ is non-singular.

[1]For completeness, we provide a proof of equivalence between the two problems in Appendix B.

III. Multi-task additive model with shared transfer functions

We now introduce our new multi-task additive model with shared transfer functions. We assume a N-task regression problem where $\{x_{ij}^{(m)}, i \in [n], j \in [p], m \in [N]\}$ are the covariates, and $\{y_i^{(m)}, i \in [n], m \in [N]\}$ denotes the response variables, where the superscript (m) is the task index. We further assume without loss of generality that the response variables have zero mean. Our multi-task model is given as follows:

$$y_i^{(m)} = \sum_{j=1}^{p}\sum_{l=1}^{L_j} \lambda_{jl}^{(m)} f_{jl}\left(x_{ij}^{(m)}\right) + \epsilon_i^{(m)} \tag{3}$$
$$\text{with } \|\boldsymbol{\lambda}_j^{(m)}\|_0 \leq 1 \text{ and } \boldsymbol{\lambda}_j^{(m)} \geq \mathbf{0} \text{ for all } j \in [p], m \in [N].$$

Note that, in our new model, the response variables are *weighted* linear combinations of p transfer functions, each of which is selected from the set $\mathcal{F}_j \triangleq \{f_{jl}, l \in [L_j]\}$ which contains L_j candidate transfer functions that model the effects of the covariate j. The ℓ_0 norm constraint on the weights $\boldsymbol{\lambda}_j^{(m)}$ prevents two transfer functions from the set $\mathcal{F}_j$ to be active for the same task. Hence, only one transfer function captures the effect of a covariate in a response variable. This constraint is crucial, as it disallows the creation of "new" transfer functions from the candidate ones by linearly combining them. While the transfer functions f_{jl} are *common* to all the tasks, the non-negative weights $\lambda_{jl}^{(m)}$ are *task-specific* and permit to scale the transfer functions specifically for each task. This offers extra flexibility as a wide range of tasks can be modeled using the model in Eq. (3) while keeping the number of (standardized) candidate transfer functions small. As we will see in Sec. VI, the non-negativity constraint in Eq. (3) facilitates the interpretation of the activation of the same transfer functions across different tasks as commonality; without this constraint, the same transfer functions could represent exactly opposite effects, e.g., higher temperatures leading to higher electricity demand for one task, and leading to lower demand for another one. Our multi-task model is illustrated in Fig. 2.

It should be noted that the sparsity level of 1 is an important component in our multi-task extension of additive models. In particular, we model each task using a linear combination of transfer functions taken from the set of candidate transfer functions $\mathcal{F}_j$. The 1-sparsity requirement ensures that the transfer functions that model each task are really *selected* (in contrast to *created*) from the candidate set and prevents the creation of new transfer functions that are linear combinations of ones from the candidate set. By removing this assumption, linear combinations of the existing transfer functions would result in cancellations and intertwining of different candidate functions which are hard to track and greatly hurt the interpretability of the proposed model.

Similarly to what is done with single-task additive models (Sec. II-B), we model transfer functions using smoothing splines. Specifically, we write:

$$f_{jl}(z) = \mathbf{s}_j(z)^T\boldsymbol{\beta}_{jl}, \tag{4}$$

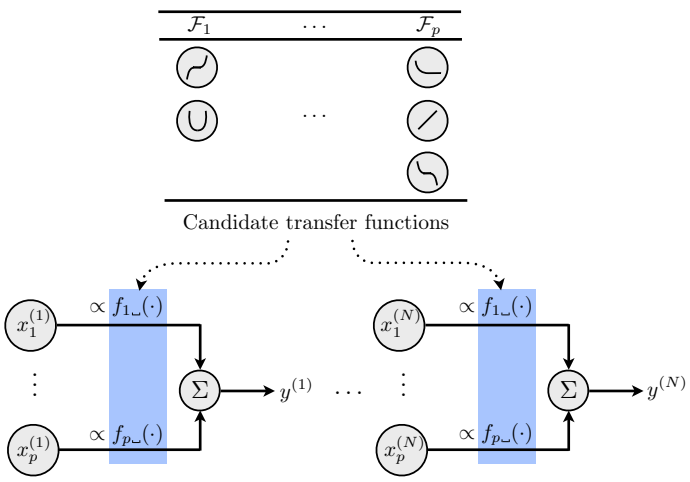


Fig. 2. Multi-task additive model diagram. $\mathcal{F}_j$ denotes the set of L_j candidate transfer functions that model the effects of covariate j. The sets $\mathcal{F}_j, 1 \leq j \leq p$ are common to all tasks. Each task is modeled as a linear combination of transfer functions chosen from the sets $\mathcal{F}_1, \ldots, \mathcal{F}_p$. In this diagram, the symbol $\sqcup$ denotes an arbitrary index in $[L_j]$.

where $\mathbf{s}_j$ and $\boldsymbol{\beta}_{jl}$ denote the centered spline basis functions and coefficients, respectively. Using this representation, we rewrite the model in Eq. (3) in the following vector form:

$$\forall m \in [N], \quad \mathbf{y}^{(m)} = \sum_{j=1}^{p} \mathbf{S}_j^{(m)} \mathbf{B}_j \boldsymbol{\lambda}_j^{(m)} + \boldsymbol{\epsilon}^{(m)},$$

where $\mathbf{S}_j^{(m)} = \left[\mathbf{s}_j\left(x_{1j}^{(m)}\right) \quad \ldots \quad \mathbf{s}_j\left(x_{nj}^{(m)}\right)\right]^T$, $\mathbf{B}_j = \left[\boldsymbol{\beta}_{j1} \quad \ldots \quad \boldsymbol{\beta}_{jL_j}\right]$, and $\boldsymbol{\epsilon}^{(m)}$ is a Gaussian iid random vector with zero mean. The model fitting then consists in finding admissible $\{\mathbf{B}_j\}_{j=1}^p$ and $\{\boldsymbol{\lambda}_j^{(m)}\}_{j\in[p],m\in[N]}$ that minimize the sum of squared residuals, while avoiding overfitting. We therefore write the problem as follows:

$$\text{(P)}: \min_{\mathbf{B}_j, \boldsymbol{\lambda}_j^{(m)}} \sum_{m=1}^{N} \left\| \mathbf{y}^{(m)} - \sum_{j=1}^{p} \mathbf{S}_j^{(m)} \mathbf{B}_j \boldsymbol{\lambda}_j^{(m)} \right\|_2^2 + \Omega(\{\mathbf{B}_j\}_{j=1}^p),$$
$$\text{subject to } \|\boldsymbol{\lambda}_j^{(m)}\|_0 \leq 1 \text{ and } \boldsymbol{\lambda}_j^{(m)} \geq \mathbf{0} \text{ for all } j, m,$$

where Ω is a regularization term that prevents model overfitting. Note that, unlike traditional additive model learning, the above problem has two types of unknowns, that is, weights and transfer functions. In this paper, we use the following regularization function

$$\Omega(\{\mathbf{B}_j\}_{j=1}^p) = \mathbf{b}^T \boldsymbol{\Sigma} \mathbf{b}, \tag{5}$$

with the regularization matrix $\boldsymbol{\Sigma} = \nu \mathbf{I}$ and $\nu > 0$, and $\mathbf{b}$ is the vector formed by concatenating $\text{vec}(\mathbf{B}_j), 1 \leq j \leq p$. Note that this regularizer penalizes large coefficients of smoothing splines with the strength of this effect tuned by ν. Table I provides a summary of the notations used in the model, as well as the matrix sizes.

The fitting problem (P) is inherently related to *sparse dictionary learning* [5] where the goal is to find the dictionary $\mathbf{D}$ and sparse codes $\mathbf{C}$ that minimize

$$\|\mathbf{Y} - \mathbf{D}\mathbf{C}\|_F^2 \text{ subject to } \|\mathbf{c}\|_0 \leq p \text{ for all } \mathbf{c} \in \mathbf{C},$$

with $\mathbf{Y} = [\mathbf{y}^{(1)} \ldots \mathbf{y}^{(N)}] \in \mathbb{R}^{n \times N}$, and p is the desired level of sparsity. To simplify the exposition of the analogy, let us consider the multi-response scenario where covariates are equal across tasks ($x_{ij}^{(m)} = x_{ij}$ for all m). In this case, we have $\mathbf{S}_j^{(m)} = \mathbf{S}_j$ for all $m \in [N]$. We define the subdictionaries (or *blocks*) $\mathbf{D}_j \triangleq \mathbf{S}_j \mathbf{B}_j$ and the global dictionary $\mathbf{D} \triangleq \begin{bmatrix}\mathbf{D}_1 & \ldots & \mathbf{D}_p\end{bmatrix}$. The problem (P) can be rewritten as follows:

$$\|\mathbf{Y} - \mathbf{D}\boldsymbol{\Lambda}\|_F^2 + \Omega(\{\mathbf{B}_j\}_{j=1}^p)$$
$$\text{subject to } \|\boldsymbol{\lambda}_j^{(m)}\|_0 \leq 1 \text{ for all } j, m \text{ and } \boldsymbol{\Lambda} \geq \mathbf{0},$$

with

$$\boldsymbol{\Lambda} = \begin{bmatrix} \boldsymbol{\lambda}_1^{(1)} & \ldots & \boldsymbol{\lambda}_1^{(N)} \\ \vdots & \vdots & \vdots \\ \boldsymbol{\lambda}_p^{(1)} & \ldots & \boldsymbol{\lambda}_p^{(N)}. \end{bmatrix}$$

Hence, the difference between sparse dictionary learning and problem (P) essentially lies in the underlying *sparsity constraints*: while in the former one the only constraint is that sparse codes have no more than p nonzero entries, in the latter they are further constrained *to have at most one nonzero entry for each subdictionary*[2]. Based on this analogy, we introduce in the next section a novel algorithm for efficiently approximating the solution of problem (P).

IV. Learning algorithm

The problem of dictionary learning has proved challenging. In fact, even if the dictionary is known, it can be NP-hard to represent a vector as a linear combination of the columns in the dictionary [23]. Problem (P) inherits the difficulty of dictionary learning, and we therefore propose an approximate alternating algorithm that solves successively for the weights $\{\boldsymbol{\lambda}_j^{(m)}\}$ and spline coefficients $\{\mathbf{B}_j\}$.

A. Weights update

We assume that the spline coefficients matrices $\{\mathbf{B}_j\}$ are given, and we define $\mathbf{D}_j^{(m)} = \mathbf{S}_j^{(m)} \mathbf{B}_j \in \mathbb{R}^{n \times L_j}$. We define the columns of each subdictionary $\mathbf{D}_j^{(m)} = \left[\mathbf{d}_{j,1}^{(m)} \quad \ldots \quad \mathbf{d}_{j,L_j}^{(m)}\right]$ to be the *atoms* of $\mathbf{D}_j^{(m)}$. Hence, an atom $\mathbf{d}_{j,l}^{(m)}$ is obtained by applying the transfer function f_{jl} to all the observations of the jth covariate: $\mathbf{d}_{j,l}^{(m)} = \left[f_{jl}(x_{1j}^{(m)}) \quad \ldots \quad f_{jl}(x_{nj}^{(m)})\right]^T$. The weight estimation problem is given by

$$\min_{\{\boldsymbol{\lambda}_j^{(m)}\}_{j,m}} \sum_{m=1}^{N} \left\| \mathbf{y}^{(m)} - \sum_{j=1}^{p} \mathbf{D}_j^{(m)} \boldsymbol{\lambda}_j^{(m)} \right\|_2^2$$
$$\text{subject to } \|\boldsymbol{\lambda}_j^{(m)}\|_0 \leq 1 \text{ and } \boldsymbol{\lambda}_j^{(m)} \geq \mathbf{0} \text{ for all } j, m.$$

[2]Note that there are other differences between the two problems. Namely, in dictionary learning, atoms are usually unconstrained unit-norm vectors, while in our model they are constrained to be linear combinations of B-splines. Moreover, our problem involves an additional regularization function.

TABLE I
SUMMARY OF NOTATIONS FOR THE PROPOSED MODEL

Notation	Meaning	Type
p	Number of covariates	Scalar
n	Number of observations	Scalar
N	Number of tasks	Scalar
L_j	Number of candidate transfer functions modeling covariate j	Scalar
T_j	Number of basis splines for covariate j	Scalar
f_{jl}	Candidate transfer function l for covariate j	Function
$\mathbf{y}^{(m)}$	Response variable (task m)	Vector of size n
$x_{ij}^{(m)}$	Observed covariate (covariate j, task m, observation i).	Scalar
$\mathbf{S}_j^{(m)}$	Spline basis matrix (covariate j, task m).	Matrix of size $n \times T_j$
$\mathbf{B}_j$	Spline coefficient matrix (covariate j)	Matrix of size $T_j \times L_j$
$\boldsymbol{\lambda}_j^{(m)}$	Weight that scale transfer function (covariate j, task m)	Vector of size L_j
$\boldsymbol{\epsilon}^{(m)}$	Noise vector (task m)	Vector of size n
ν	Regularization parameter	Scalar
$\mathbf{D_j}$	Subdictionary $\mathbf{D}_j \triangleq \mathbf{S}_j\mathbf{B}_j$	Matrix of size $n \times L_j$
$\mathbf{D}$	Global dictionary $\mathbf{D} = [\mathbf{D}_1 \dots \mathbf{D}_p]$	Matrix of size $n \times (\sum_j L_j)$.

The above problem can be seen as computing the best non-negative p-sparse approximations of the signals $\mathbf{y}^{(m)}$ in the dictionary $\mathbf{D}^{(m)} = [\mathbf{D}_1^{(m)}|\dots|\mathbf{D}_p^{(m)}]$, provided that no two active dictionary atoms belong to the same subdictionary. Note that the non-negativity constraint is important for interpretability, as it guarantees a similar interpretation for all tasks activating a similar transfer function. We first note that this problem is separable and therefore can be solved independently for each task. Next, we simplify the problem and drop the non-negativity constraints on $\boldsymbol{\lambda}_j^{(m)}$. Following the approach used in [24], [25], non-negative coefficients can then be obtained in a post-processing step by including the negative of each atom in the dictionary[3], as we have:

$$\sum_{j=1}^{p} \mathbf{D}_j^{(m)}\boldsymbol{\lambda}_j^{(m)} = \sum_{j=1}^{p} \begin{bmatrix}\mathbf{D}_j^{(m)} & -\mathbf{D}_j^{(m)}\end{bmatrix} \begin{bmatrix}\max(0, \boldsymbol{\lambda}_j^{(m)}) \\ \max(0, -\boldsymbol{\lambda}_j^{(m)})\end{bmatrix}.$$

For a single task, our weight estimation problem is written:

$$\min_{\{\boldsymbol{\lambda}_j\}_{j=1}^p} \left\| \mathbf{y} - \sum_{j=1}^{p} \mathbf{D}_j\boldsymbol{\lambda}_j \right\|_2^2$$
$$\text{subject to } \|\boldsymbol{\lambda}_j\|_0 \leq 1 \text{ for all } j \in [p].$$

To solve this problem, we propose the iterative algorithm Block Constrained Orthogonal Matching Pursuit (BC-OMP). It is an extension of the popular Orthogonal Matching Pursuit algorithm [7], [8] which is an efficient greedy method for solving sparse coding problems. At each iteration of the algorithm, we select the dictionary atom which has the strongest correlation with the residual, provided it belongs to an available subdictionary whose index is listed in $\mathcal{A}_{j-1}$. The residual is then updated using an orthogonal projection onto the selected atoms. The availability set $\mathcal{A}_j$ is in turn updated to prevent selecting two atoms from the same subdictionary. Details of our approach are presented in Algorithm 1.

[3]This post-processing step results in doubling the number of candidate transfer functions. That is, for covariate j, we have $2L_j$ candidate transfer functions, as we consider the positive and negative versions of every transfer function. Note that, with this post-processing step, we are guaranteed to reach a solution that is at least as good as the one obtained with L_j transfer functions for covariate j. We stress however that this step is an *approximation*, as we generally do not reach the solution to the problem with $2L_j$ transfer functions.

Algorithm 1 BC-OMP

Input: Subdictionaries $\mathbf{D}_1, \dots, \mathbf{D}_p$, signal $\mathbf{y}$.
Output: Weight vectors $\boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_p$.

Initialization:
Available covariates: $\mathcal{A}_0 \leftarrow \{1, \dots, p\}$, residual: $\mathbf{r}_0 \leftarrow \mathbf{y}$, selected atoms: $\mathbf{U}_0 \leftarrow \emptyset$, weight vectors: $\boldsymbol{\lambda}_j \leftarrow \mathbf{0}$ for all $j \in [p]$.
for all $j = 1, \dots, p$ **do**
Selection step:

$$\{k_j, l_j\} \leftarrow \underset{k \in \mathcal{A}_{j-1}}{\operatorname{argmax}} \ \underset{l \in \{1,\dots,L_k\}}{\operatorname{argmax}} \ \frac{|\langle \mathbf{r}_{j-1}, \mathbf{d}_{k,l}\rangle|}{\|\mathbf{d}_{k,l}\|_2}.$$

Update step:

$$\mathcal{A}_j \leftarrow \mathcal{A}_{j-1}\backslash\{k_j\}, \mathbf{U}_j \leftarrow \begin{bmatrix}\mathbf{U}_{j-1} & \{\mathbf{d}_{k_j,l_j}\}\end{bmatrix}$$
$$\mathbf{c}_j \leftarrow \mathbf{U}_j^{\dagger}\mathbf{y}, \mathbf{r}_j \leftarrow \mathbf{y} - \mathbf{U}_j\mathbf{c}_j.$$

end for
for all $j = 1, \dots, p$ **do**
Set $\boldsymbol{\lambda}_{k_j}[l_j] \leftarrow \mathbf{c}_p[j]$.
end for

B. Spline coefficients update

We now solve the problem of learning the spline coefficients $\mathbf{B}_j$ given the fixed weights $\boldsymbol{\lambda}_j^{(m)}$. We note that:

$$\sum_{j=1}^{p} \mathbf{S}_j^{(m)}\mathbf{B}_j\boldsymbol{\lambda}_j^{(m)} = \sum_{j=1}^{p}((\boldsymbol{\lambda}_j^{(m)})^T \otimes \mathbf{S}_j^{(m)})\text{vec}(\mathbf{B}_j)$$
$$= \begin{bmatrix}(\boldsymbol{\lambda}_1^{(m)})^T \otimes \mathbf{S}_1^{(m)} & \dots & (\boldsymbol{\lambda}_p^{(m)})^T \otimes \mathbf{S}_p^{(m)}\end{bmatrix} \begin{bmatrix}\text{vec}(\mathbf{B}_1) \\ \vdots \\ \text{vec}(\mathbf{B}_p)\end{bmatrix}$$
$$\triangleq \mathbf{Z}^{(m)}\mathbf{b}.$$

Thus, the objective function becomes:

$$\sum_{m=1}^{N} \|\mathbf{y}^{(m)} - \mathbf{Z}^{(m)}\mathbf{b}\|_2^2 + \mathbf{b}^T\boldsymbol{\Sigma}\mathbf{b} = \|\text{vec}(\mathbf{Y}) - \mathbf{Z}\mathbf{b}\|_2^2 + \mathbf{b}^T\boldsymbol{\Sigma}\mathbf{b},$$

Algorithm 2 Multi-task additive model fitting algorithm

Input: Covariates $\{x_{ij}^{(m)}\}_{i,j,m}$, response variables $\{y_i^{(m)}\}_{i,m}$, parameters $L_1, \ldots, L_p$ and ν.
Output: Spline coefficients $\{\mathbf{B}_j\}_j$, scaling weights $\{\boldsymbol{\lambda}_j^{(m)}\}_{j,m}$.

Initialize $\mathbf{B}_1, \ldots, \mathbf{B}_p$ with random entries from $\mathcal{N}(0,1)$.
while not converged **do**
  Weights update: Use BC-OMP for each response variable $\mathbf{y}^{(m)}$ to estimate $\{\boldsymbol{\lambda}_j^{(m)}\}_{j\in[p],m\in[N]}$.
  Spline coefficients update: Use Eq. (6) to update the spline coefficients.
end while
Ensure the non-negativity of the weights:

$$\mathbf{B}_j \leftarrow \begin{bmatrix}\mathbf{B}_j & -\mathbf{B}_j\end{bmatrix},$$
$$\boldsymbol{\lambda}_j^{(m)} \leftarrow \begin{bmatrix}\max(0, \boldsymbol{\lambda}_j^{(m)}) \\ \max(0, -\boldsymbol{\lambda}_j^{(m)})\end{bmatrix},$$

for all j, m.

with $\text{vec}(\mathbf{Y}) = \begin{bmatrix}\mathbf{y}^{(1)} \\ \vdots \\ \mathbf{y}^{(N)}\end{bmatrix}$ and $\mathbf{Z} = \begin{bmatrix}\mathbf{Z}^{(1)} \\ \vdots \\ \mathbf{Z}^{(N)}\end{bmatrix}$. The minimum of the above least-squares program with respect to $\mathbf{b}$ is given by:

$$\hat{\mathbf{b}} = (\mathbf{Z}^\mathbf{T}\mathbf{Z} + \boldsymbol{\Sigma})^{-1}\mathbf{Z}^\mathbf{T}\text{vec}(\mathbf{Y}). \tag{6}$$

Given the spline basis coefficients $\mathbf{B}_j$, the transfer functions can then be obtained by multiplying the obtained coefficients with the spline basis vectors (Eq. (4)).

It should be noted that the spline coefficients update procedure is related to that of MOD algorithm [26], which updates the dictionary atoms along the optimal directions. However, our update equation (Eq. (6)) is slightly more complex due to the subdictionaries structure in our problem.

C. Complete learning algorithm

The complete algorithm is shown in Algorithm 2. Using a random initialization of the spline coefficients, we iterate through the weights update and spline coefficients update steps, until a termination criterion is met. In this paper, we terminate the algorithm after a fixed number of iterations. In a final step, the matrices $\mathbf{B}_j$ and $\boldsymbol{\lambda}_j^{(m)}$ are modified to ensure the non-negativity of the weights, as discussed in Section IV-A.

We briefly analyze the computational complexity of Algorithm 2. We assume for simplicity that $L_j = L$ for all $j \in [p]$. The BC-OMP algorithm involves a selection and an update step whose complexity are $O(Lpn)$ and $O(pn + p^2)$ respectively. Assuming that $p < n$, the selection step dominates and the overall complexity of BC-OMP is $O(Lp^2n)$. Doing this operation for each task results in a complexity of $O(nNLp^2)$. The spline coefficients update involves the computation in Eq. (6). To compute the complexity of this operation, note that $\mathbf{Z}$ has nN rows and LTp columns (where T is the number of spline basis functions, or equivalently the number of columns of $\mathbf{S}_j^{(m)}$, assumed to be equal for all covariates j for simplicity). For typical problems, this matrix is tall and the complexity is driven by the computation of $\mathbf{Z}^T\mathbf{Z}$, which is of complexity $O(nN(LTp)^2)$. Hence, assuming that we run Algorithm 2 for a fixed number of iterations, the complexity of our overall algorithm is $O(nN(LTp)^2)$.

Our algorithm is therefore *linear* in the number of tasks N, and dimension n, while being quadratic with respect to the number of candidate transfer functions per task L. In comparison, learning an additive model independently for each task has complexity $O(nN(Tp)^2)$. Compared to the independent additive model approach, the price to pay of our algorithm is therefore L^2, which remains small in most problems of interest.

V. Recovery condition for BC-OMP

We analyze in this section the weights update algorithm BC-OMP. While BC-OMP represents one building block of the global algorithm (Algorithm 2), an analysis of the recovery conditions of BC-OMP is important as it provides insights onto the success of our algorithm.

We suppose that $\mathbf{y}$ is a superposition of p elements in $\mathbf{D} = [\mathbf{D}_1|\ldots|\mathbf{D}_p]$ such that no two active elements belong to the same subdictionary $\mathbf{D}_j$, i.e., $\mathbf{y} = \sum_{j=1}^p \gamma_j \mathbf{d}_{j,l_j}$. For simplicity, we further assume that the atoms $\mathbf{d}_{j,l_j}$ are linearly independent and the γ_j are all nonzero[4]. We develop a sufficient condition for the recovery of the correct atoms using BC-OMP.

We first note that the difference between OMP and BC-OMP algorithms lies in their search space: while OMP selects atoms from the dictionary $\mathbf{D}$ having maximal inner product with the residual, BC-OMP further imposes a constraint that the selected atom belong to an available subdictionary where no atoms have been previously selected. It follows that if OMP succeeds in the recovery of the correct atoms of $\mathbf{y}$, the same holds for BC-OMP. Therefore, any condition that guarantees the recovery of OMP is *a fortiori* a recovery condition for BC-OMP. Many OMP recovery conditions have been proposed in the literature (see e.g., [27], [28]). The following theorem in [27] gives a popular and practical recovery condition of OMP for the global dictionary $\mathbf{D}$:

Theorem 1 ([27])**.** *Let*

$$\mu \triangleq \max_{\substack{\mathbf{d},\mathbf{d}'\in\mathbf{D}\\ \mathbf{d}\neq\mathbf{d}'}} |\langle \mathbf{d}, \mathbf{d}'\rangle|.$$

OMP recovers every superposition of p atoms from $\mathbf{D}$ whenever the following condition is satisfied:

$$p < \frac{1}{2}\left(\mu^{-1} + 1\right). \tag{7}$$

The quantity μ, called *coherence*, measures the similarity between dictionary atoms. The values of μ that are close to 1 may violate the recovery condition in Eq. (7), thus leaving us without any guarantee that OMP or BC-OMP will recover the correct atoms. Unfortunately enough, in our multi-task learning framework, μ is typically close to 1. To see this, note

[4]Otherwise, the signal has a representation with fewer atoms and one can remove the unused covariates $j \in [p]$.

that the inner product between two atoms of the same subdictionary is equal to $\langle \mathbf{d}_{j,l}, \mathbf{d}_{j,l'} \rangle = \sum_{i=1}^{n} f_{jl}(x_{ij}) f_{jl'}(x_{ij})$. In practice, transfer functions in the same subdictionary often bear strong resemblance, e.g., similar monotic behavior (see Sec. VI for examples). Thus, $f_{jl} \approx f_{jl'}$ and $|\langle \mathbf{d}_{j,l}, \mathbf{d}_{j,l'} \rangle| \approx 1$, which leads to a large coherence value. On the other hand, the inner product of atoms from different subdictionaries $|\langle \mathbf{d}_{j,l}, \mathbf{d}_{j',l'} \rangle| = \sum_{i=1}^{n} f_{jl}(x_{ij}) f_{j'l'}(x_{ij'})$ is close to zero when the covariates j and j' are "sufficiently independent".[5] To circumvent the above violation of the recovery condition in Theorem 1 due to the large *global coherence* of the dictionary, we first define coherence within and across subdictionaries:

Definition 1.

$$\mu_{intra} \triangleq \max_{j \in [p]} \max_{\substack{(l,l') \in [L_j] \\ l \neq l'}} |\langle \mathbf{d}_{j,l}, \mathbf{d}_{j,l'} \rangle| ,$$

$$\mu_{inter} \triangleq \max_{\substack{(j,j') \in [p] \\ j \neq j'}} \max_{(l,l') \in [L_j] \times [L_{j'}]} |\langle \mathbf{d}_{j,l}, \mathbf{d}_{j',l'} \rangle| .$$

Using these definitions, we derive the following recovery condition for BC-OMP:

Theorem 2 (Recovery condition)**.** *If the following condition holds:*

$$\mu_{intra} + 2(p-1)\mu_{inter} < 1,$$

then BC-OMP recovers the correct atoms and their coefficients.

The detailed proof can be found in the Appendix. Unlike the recovery condition in Theorem 1, Theorem 2 does not depend on the global coherence of the dictionary. This recovery condition is particularly interesting in our applications where μ_{intra} typically takes large values due to strong resemblance between the transfer functions in the same subdictionary, while μ_{inter} is small when the covariates are sufficiently statistically independent. Interestingly, Theorem 2 shows that, in the special case where subdictionaries are orthogonal to each other, the parameter μ_{intra} can take values arbitrarily close (but not equal) to 1 and BC-OMP still succeeds in the recovery. In contrast, the recovery condition of Theorem 1 is not satisfied in this case since the global coherence μ is close to 1. We also mention that our recovery condition is not limited to BC-OMP. In fact, it is also valid for the OMP algorithm, provided the signal $\mathbf{y}$ follows the above sparsity structure (i.e., $\mathbf{y}$ is a superposition of p elements, each taken from a different subdictionary). Hence, Theorem 2 is important in its own right as it provides a novel recovery condition for OMP for an interesting class of sparsity structures.

Finally, we draw the reader's attention to some results related to the proposed recovery condition in Theorem 2. In [29], the authors provide a new analysis of Matching Pursuit, when the dictionary is built from an incoherent union of possibly coherent subdictionaries. A sufficient condition that guarantees the selection of atoms from the correct subdictionaries is shown. In other words, the exact recovery of atoms is dropped and a sufficient condition for the weaker subdictionary recovery property is shown. This is completely different from our setting, where we require the correct atoms to be recovered when the signal contains at most one atom per subdictionary. In [30], the "block sparse" model is introduced: the signals' non-zero entries appear in blocks rather than being spread throughout the vector. Coherence-based recovery conditions for a block version of OMP are shown. Once again, our model significantly differs from this one, as we assume *one* active component per subdictionary (or block), whereas the work of [30] assumes that nonzero entries occur in clusters.

[5]More precisely, if we model the covariates as random variables X_j and $X_{j'}$, then $1/n \sum_{i=1}^{n} f_{jl}(x_{ij}) f_{j'l'}(x_{ij'})$ can be seen as a sample estimate of the population covariance $\mathbb{E}[f_{jl}(X_j) f_{j'l'}(X_{j'})]$ which will be 0 if X_j and $X_{j'}$ are independent. Note that in this argument we use the fact that the transfer functions are centered (see Sec. II).

VI. Experimental results

In this section, we present experimental results. Baseline methods and practical implementation details of our algorithm are explained in Sec. VI-A. Sec. VI-B reports results on synthetic data, and the following two sections show results on electric load forecasting problems. More background on using additive models for electricity demand forecasting can be found in [31], [32] for example.

A. Experimental setup

We compare the proposed multi-task learning approach to the following baseline regression methods:

1) **Linear Regression (LR):** A linear regressor is learned independently using ϵ-SVR [33] with a linear kernel for each task. The penalty parameter C is set using a cross-validation procedure for each task. We use the Liblinear implementation [34].
2) **Support Vector Regression with Radial Basis Function kernel (SVR-RBF):** We learn an ϵ-SVR-RBF regressor independently for each task where the penalty parameter C and kernel bandwidth σ are determined using cross-validation. We use the LibSVM implementation [35].
3) **Independent Additive Models (IAM):** An additive model is fitted for each task independently using the `mgcv` package in `R` [2].
4) **K-Means and Additive Model (KAM):** This is a two-step approach, where in the first step we use the K-means algorithm to group the set of tasks into different clusters. In the second step, one additive model is learned independently for each cluster centroid. The prediction of a signal is given by the prediction for the centroid of the cluster it belongs to.

We now discuss practical aspects of our algorithm. For simplicity, we have chosen the regularization value $\nu = 1$ in all our experiments (see Eq. (5)). Note that a cross-validation procedure is likely to give better results, but is more computationally expensive. Moreover, in all experiments we set the parameters $L_j = L$ for all covariates $j \in [p]$. We envision that, in real-world applications, the parameters L_j will be manually selected by domain experts (possibly in an iterative procedure) to find the optimal trade-off between predictive performance and model interpretability from their

point of view. In the experiments below, we show results for different values of L to evaluate how the choice of L_j affects the predictive performance. Finally, similarly to K-means, our proposed algorithm can incur the problem of "empty clusters" when the number L of transfer functions per covariate is large. We circumvent this problem by checking at each iteration for unused transfer functions and, when such a transfer function is detected, replacing it with the transfer function that leads to minimum error for the task with the currently highest approximation error.

B. Synthetic experiment

In our first experiment, we generate $n = 100$ samples according to the multi-task additive model in Eq. (3), with $p = 10$ covariates, $N = 200$ tasks and $L_1 = \cdots = L_p = L = 3$ candidate transfer functions per covariate. For simplicity, we take the covariates $x_{ij}^{(m)}$ to be equal for all tasks (i.e., $x_{ij}^{(m)} = x_{ij}$ for all $m \in [N]$), and randomly sample x_{ij} from the uniform distribution in $[-1, 1]$. In this synthetic example, the ground truth transfer functions f_{jl} are randomly generated smooth functions, and the scaling weights are chosen to be non-negative random numbers. Finally, the model noise $\epsilon_i^{(m)}$ is iid, and follows the standard normal distribution.

We first assess the quality of the estimated transfer functions using our algorithm, and compare it to the ground truth transfer functions (known in this synthetic setting), as well as the functions estimated with IAM method (which treats each task independently). For a fixed covariate, we show in Fig. 3 (a-c) the $L = 3$ associated transfer functions, as well as the proposed and IAM estimations. Clearly, the estimation of the true transfer functions using our multi-task approach is much more accurate and resilient to noise than IAM. In fact, the true and estimated functions using our approach nearly coincide, despite the fact that observations are highly noisy, and the relatively low sample size. We then compare the prediction performance (in terms of Root Mean Squared Error – RMSE) of the proposed method to other competitor methods on a test set of 400 samples. The results are illustrated in Fig. 4. The proposed approach leads to significantly lower RMSE compared to other approaches on this synthetic example. Despite the limited number of training data, our approach correctly detects and leverages the correlation between the different tasks to significantly improve the results.

To emphasize the importance of the model introduced in (P), and in particular the proposed sparsity constraint, we evaluate the correctness of the supports obtained using the proposed algorithm. In our setting of additive model regression, the supports of sparse codes $\boldsymbol{\lambda}_j^{(m)}$ denote the associations between the tasks and the transfer functions, and are therefore particularly important to recover correctly. We compare the proposed approach, in terms of support recovery, with one that replaces BC-OMP with a traditional OMP sparse coding step (using a sparse constraint $\|\cdot\|_0 \leq p$).[6] We show in Fig. 5 the percentage of tasks that use the incorrect transfer function, for the different covariates. Interestingly, using OMP instead of BC-OMP induces a large loss in the support recovery results, which is due to a wrong choice of the transfer functions, or to the absence of choice of any transfer functions for a particular covariate. The proposed sparsity constraint is particularly adapted to additive models being studied in this paper, as they impose structure on the transfer functions which helps interpretability and the effective recovery even under noisy conditions.

[6]Note that such an experiment is possible here as we have the ground truth information.

C. Modeling of smart meter data

In this experiment we use data from a smart metering trial of the Irish Commission for Energy Regulation (CER) [36]. The data set contains half-hourly electricity consumption data from July 14, 2009 to December 31, 2010 for approx. $5,000$ residential (RES) and small-to-medium enterprise (SME) customers. It comes with survey information about different demographic and socio-economic indicator, e.g., number of people living in the household, type of appliances, and business opening times. In our experiment, we only use customers that do not have any missing consumption data, leaving us with a total of $N = 4,066$ meters out of which $3,639$ are residential and 427 SME customers. Since half-hourly smart meter data is very volatile due to the stochastic nature of electricity consumption at the individual household level, we aggregate each signal over 6 time points to obtain one measurement every 3 hours. We split the data into 12 months of training and 6 months of test data. We consider "Hour Of Day", "Time Of Year" and "Day of Week" as covariates.

Table II shows the average RMSE on the training and test data over all $N = 4,066$ meters. Here, we use $L = 2$ for the number of candidate transfer functions per covariate in our approach, and the number of clusters in the KAM method. In terms of predictive performance, our proposed approach clearly outperforms LR and KAM; it performs only slight worse than IAM albeit using only $L = 2$ different transfer functions per covariate while IAM learns independently one additive model per signal. Note that the methods based on additive models are competitive with SVR-RBF, while the latter approach is computationally expensive at training and test time, which makes it only moderately suitable for large-scale problems, besides leading to models that are unfortunately difficult to interpret. Fig. 6 shows the performance evolution of both the proposed method and KAM with respect to L. The performance of KAM is consistently worse than the proposed algorithm, regardless of the number of clusters. Moreover, it can be seen that both the training and testing performance of the proposed method approximately stabilize for $L \geq 3$. The choice of $L = 2$ is therefore a fair tradeoff between model interpretability and performance.

In an attempt to assess the interpretability of the different methods, Table II shows the *model complexity* of the different methods learned on the CER data. The complexity is defined

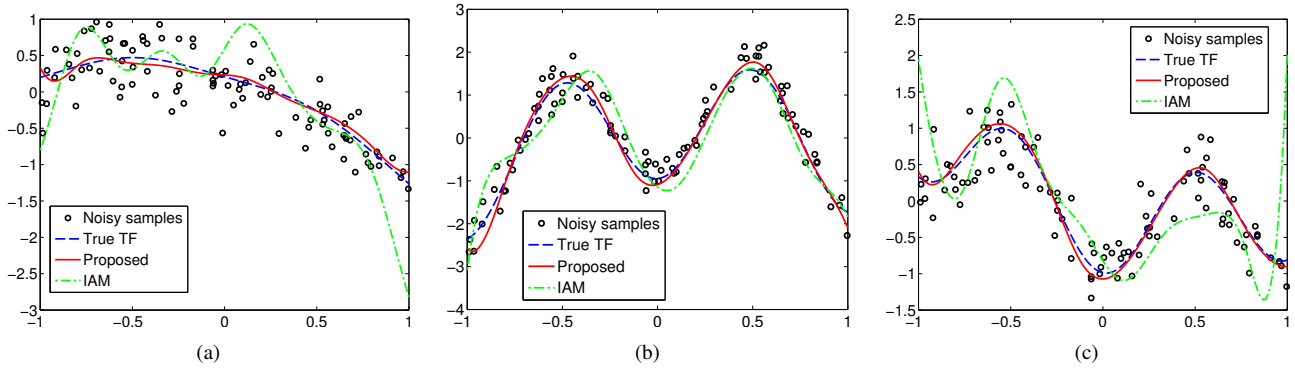


Fig. 3. Estimated transfer functions for the synthetic experiment, using the proposed and IAM methods. The black dots indicate the noisy observations used to estimate the transfer functions, and the blue dashed lines represent the *true* transfer functions. The results are shown for a *fixed covariate*, leading to the $L = 3$ estimated transfer functions that are illustrated. For the IAM method, the depicted transfer functions correspond to estimations obtained on arbitrary tasks involving the true transfer functions.

TABLE II

AVERAGE RMSE OVER ALL 4,066 TASKS IN THE CER DATA SET, AND MODEL COMPLEXITY. FOR THE PROPOSED AND KAM METHOD, THE RESULTS WITH $L = 2$ ARE SHOWN. FOR EASIER COMPARISON, THE FOURTH COLUMN SHOWS THE *normalized* MODEL COMPLEXITY, I.E., THE MODEL COMPLEXITY DIVIDED BY Np WHERE N IS THE NUMBER OF TASKS AND p THE NUMBER OF COVARIATES. T IS THE NUMBER OF ELEMENTS IN THE SPLINE BASIS (EQUAL FOR ALL COVARIATES, FOR SIMPLICITY). #SVS DENOTES THE NUMBER OF SUPPORT VECTORS FOR THE SVR-RBF MODEL.

Method	RMSE		Model complexity	
	Training	Testing	Theoretical	Numerical example
Proposed	2.6	2.7	$p(TL + 2N)$	2.01
LR	3.1	3.1	Np	1
SVR-RBF	**2.2**	**2.5**	$2\sum_{m=1}^{N} \#\{\text{SVs task } m\}$	≈ 1600
IAM	2.3	2.6	pTN	12
KAM	3.6	3.6	pTL	**0.01**

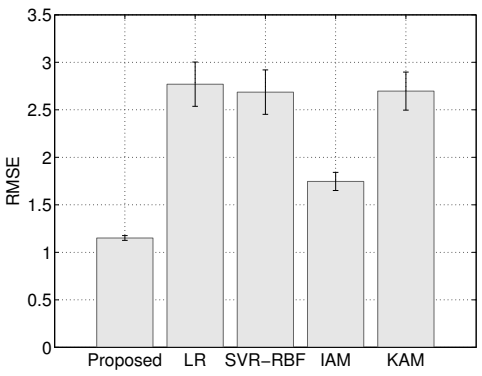


Fig. 4. Average test RMSEs across tasks of the proposed method, and the competing methods for the synthetic experiment. The experiments are performed on 50 independent trials.

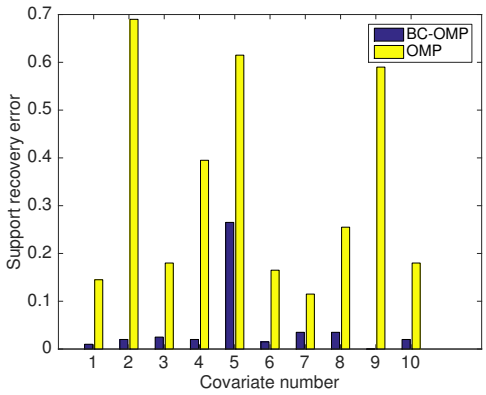


Fig. 5. Percentage of tasks that use the incorrect transfer function, for the different covariates in the toy example.

as the number of scalar variables needed to store the model.[7] For easier numerical comparison, we divide the numbers in the rightmost column of Table II by Np, i.e., the number of tasks times the number of covariates. As it can be seen, the SVR-RBF is the most complex model, since it depends on the number of support vectors (that scales linearly with the number of observations n). IAM also has a high complexity as it fits one additive model per task: it stores pT variables for each task, resulting in a complexity of pTN. On the other hand, the proposed approach has a complexity of $pTL+2pN$. In fact, the proposed model involves two distinct quantities:

[7]We emphasize that model complexity is not equivalent to the interpretability of a model, which is difficult to quantify in general. Nevertheless, we believe that in the examples in this section, complexity can be used as indication of how hard (i.e. how time-consuming and tedious) it would be for a domain expert to interpret (i.e. to inspect and grasp) all the models.

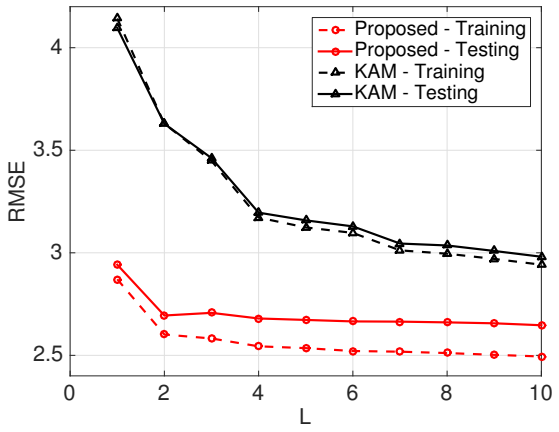


Fig. 6. Average training and testing RMSE on the CER data set vs. number of candidate functions (per covariate) and clusters L, respectively, for the proposed and the KAM method.

(i) the common transfer functions, which consist in pTL variables, and (ii) the *one-sparse* vectors $\{\boldsymbol{\lambda}_j^{(m)}\}$ that store the assignments of the candidate transfer functions to the tasks. Storing the set of vectors $\{\boldsymbol{\lambda}_j^{(m)}\}$ costs $2Np$ as the location and value of the non-zero coefficient need to be stored for each vector. It should be noted that *a scalar* is enough to indicate the activation of a stored transfer function for a given task, while IAM needs to store *functions* for each task. The results in Table II clearly confirm the advantage of our method in terms of model complexity.

Fig. 7 (a)-(c) display the transfer functions obtained by our method for $L = 2$, and Fig. 7 (d) shows the corresponding matrix of correlations $|\mathbf{D}^T\mathbf{D}|$ between the atoms of the dictionary estimated using the proposed algorithm[8]. As noted in Section V, this matrix has a block diagonal structure due to similarites between transfer functions depending on the same covariate on the one hand, and independence of different covariates on the other hand. We obtain the coherence values $\mu_{intra} = 0.87$ and $\mu_{inter} \approx 0$ (see Definition 1), satisfying our recovery condition for BC-OMP in Theorem 2, while the original condition in Theorem 1 for recovery in OMP is clearly not satisfied.

Let us consider the interpretability of the transfer functions learned using our method, and study correspondances with the customer survey information in the CER data set. Table III relates the activation of the "Hour of Day" transfer functions to the customer type (residential vs. SME). In this experiment we chose $L = 2$, resulting in 4 "final" transfer functions due to the non-negativity post-processing step discussed in Sec. IV-A. One can see that, for residential customers, overwhelmingly the first transfer function is activated, while the majority of SME signals is modeled using the second one. Looking at the shape of the transfer functions, this intuitively makes sense: the consumption of residential customers typically peaks in the evening, while SMEs consume most electricity during the day. Similarly, Table IV shows the correspondence between the activation of the "Day of Week" transfer function, and the SME business days (which is available from the CER survey information). Again, there is an intuitive and easy-to-interpret correspondence between the learned models and available ground truth information. This analysis underlines the interpretability of the proposed model, as it succeeds in associating interpretable "concepts" to the different transfer functions (e.g., residential/SME). In contrast, fitting an independent additive model would have resulted in too many transfer functions that such an association between transfer functions and concepts would be difficult to analyze and memorize.

Finally, we evaluate the performance of our method in a setting where training data is scarce. For this purpose, we consider now a training set of n samples that are randomly selected from the CER data, and consider the remaining data for testing. Fig. 8 illustrates the testing RMSE of the proposed method (with $L = 2$) and the other competing methods with respect to n. It can be seen that when the training data is scarce, the proposed method outperforms IAM, and our method inherits the advantages of traditional multi-task learning by sharing information across tasks, and hence avoids overfitting. Note that for larger n, the gap between the two methods decreases, and IAM slightly outperforms our approach (with $L = 2$), as it provides a much more flexible model, which however suffers from lack of interpretability. Note finally that our approach consistently outperforms all other competing methods (LR, SVR-RBF, KAM) in the range of training samples in Fig. 8.

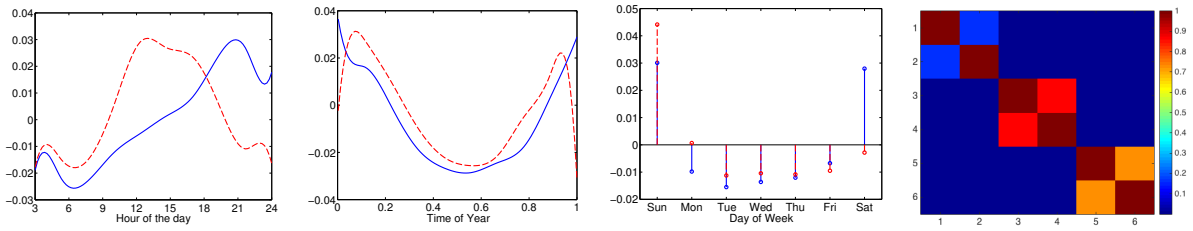
D. Intra-signal multi-task learning

In this last set of experiments, we consider another application of our multi-task learning framework. A common approach in hourly electrical load forecasting is to treat each hourly period separately and use different models for each hour of the day (see, e.g., [32] and [37]). Besides the computational burden, such approaches unfortunately fail to discover *intra-daily* commonalities in the electricity consumption during different hours. Moreover, the resulting models are difficult to interpret.

We address those issues using the proposed multi-task framework. Given a signal $\mathbf{y} \in \mathbb{R}^n$ representing hourly electrical loads, we first reshape the signal into a matrix $\mathbf{Y} \in \mathbb{R}^{d\times 24}$, where the d rows represent the days, and the columns the hours in a day. We then treat the columns of $\mathbf{Y}$ as separate tasks, and fit our model using the proposed algorithm. For this experiment, we use 4.5 years of data from the GEFCom 2012 load forecasting challenge [38], considering "Time Of Year", "Day of Week" and "Temperature" as covariates. The response variable is set to be the sum of the 20 zonal level series expressed in gigawatt. Moreover, the temperature covariates are obtained by computing the average signals over the 11 weather stations provided in the data. We use the first $d = 1,642$ days in the data set. The first 4 years of the data are considered for training, and the remaining for testing.

We compare the proposed approach to single-task regression methods that do not split the signal into hourly signals. Specif-

[8]In this experiment, the observed covariates are equal for all the tasks (i.e., $x_{ij}^{(m)} = x_{ij}$), since Hour of day, Time of year and Day of week are clearly independent of the task at hand. This leads to a unique dictionary $\mathbf{D}$ that is independent of the task. See Sec. III for further details.



(a) (b) (c) (d)

Fig. 7. Transfer functions obtained with $L = 2$ (a-c), and correlation between atoms (d). (a): Hour of Day, (b): Time of Year (0: January, 1st, 1: December, 31st), (c): Day of Week.

TABLE III
PERCENTAGE OF THE ACTIVATION OF "HOUR OF DAY" TRANSFER FUNCTIONS FOR RESIDENTIAL AND SME CUSTOMERS.

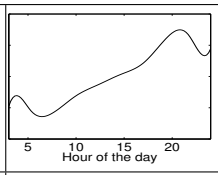
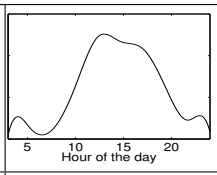
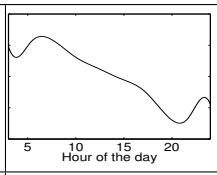
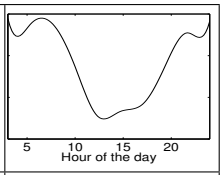
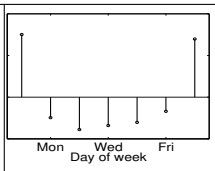
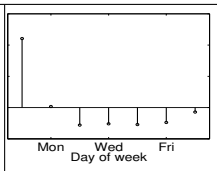
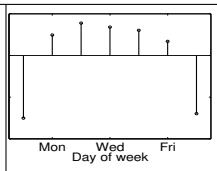
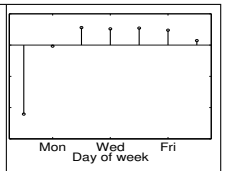
				
Residential	**89%**	9%	0%	2%
SME	19%	**68%**	8 %	5 %

TABLE IV
PERCENTAGE OF THE ACTIVATION OF "DAY OF WEEK" TRANSFER FUNCTIONS FOR SMES WITH DIFFERENT BUSINESS DAYS (SEE THE LEFT COLUMN).

				
Week days only	1%	2%	**93%**	4%
Week days + Saturday	0%	0%	37%	**63%**
All days	36%	5%	19%	**40%**

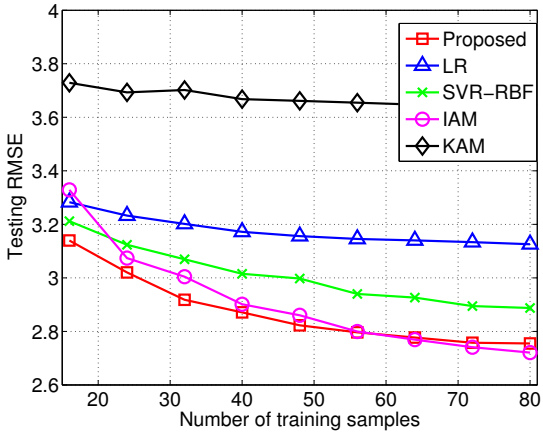


Fig. 8. Testing RMSE versus number of training samples for the CER experiment. For $n \leq 50$, the Wilcoxon statistical test shows that the proposed method outperforms the other methods, with a significance level of 0.05.

ically, considering an additional "Hour of Day" covariate, we fit linear and nonlinear models using LR, SVR-RBF as well as an Additive Model that reads

$$y_i = f_1(\text{Hour of Day}_i) + f_2(\text{Time of Year}_i) + f_3(\text{Day of Week}_i) + f_4(\text{Temperature}_i).$$

In addition, we compare the proposed approach to IAM using the same split into hourly signals.

Fig. 9 illustrates the evolution of the training and testing RMSE of the proposed method with respect to the parameter L for the GEFCom task. Based on the evolution of the training

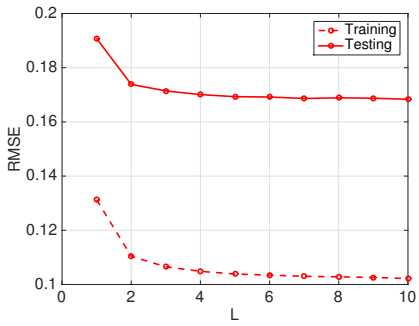


Fig. 9. Evolution of the training and testing performance of the proposed method with respect to the parameter L for the GEFCom task.

TABLE V
TRAINING AND TESTING RMSE FOR THE GEFCOM TASK.

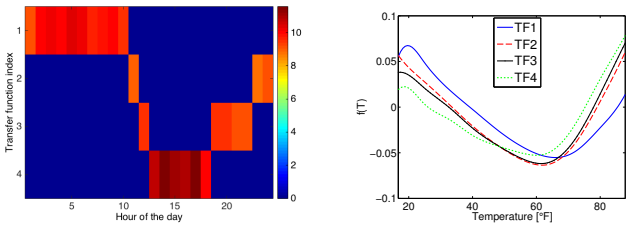
Method	RMSE training	RMSE testing
Proposed ($L = 4$)	**0.11**	**0.17**
LR	0.62	0.50
SVR-RBF	**0.11**	0.18
Additive Model (AM)	0.13	0.19
IAM	**0.11**	0.18

performance with respect to L, we choose to set $L = 4$ in the experiments in order to achieve a good tradeoff between model interpretability and performance. Table V shows the result of the comparison. It can be seen that, with $L = 4$, the proposed approach outperforms all other methods in terms of testing RMSE. By splitting the signal into hourly signals, our algorithm yields a testing RMSE that has improved roughly by 10% with respect to AM. In addition, our approach slightly outperforms IAM in terms of testing accuracy in this experiment, while learning much less transfer functions. The paired Wilcoxon test shows that the improvement of the proposed method over IAM is statistically significant at a significance level of 0.05.

Interestingly, our algorithm yields a clustering of the hours of the day with some intuitive interpretations. To visualize such information, we consider the matrix $\mathbf{\Lambda}_{\text{temp}}$ which shows the assignment of the $L = 4$ temperature transfer functions (displayed in Fig. 10 (b)) to the 24 signals representing different hours per day. To be more specific, the matrix $\mathbf{\Lambda}_{\text{temp}}$ is given by

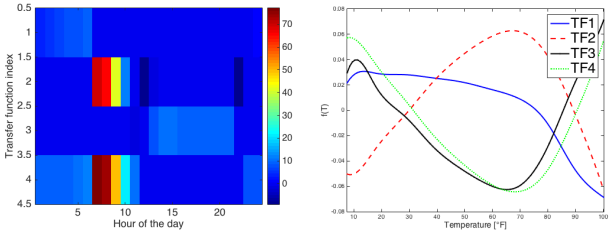
$$\mathbf{\Lambda}_{\text{temp}} = \begin{bmatrix} \lambda_{j1}^{(1)} & \dots & \lambda_{j1}^{(24)} \\ \vdots & \vdots & \vdots \\ \lambda_{j4}^{(1)} & \dots & \lambda_{j4}^{(24)} \end{bmatrix},$$

where j corresponds to the index of the "Temperature" covariate. This correspondence matrix is sparse, as it contains at most one nonzero coefficient per column, and it shows the transfer function to which each task is assigned. Such a matrix therefore provides a practical visualization of the tasks that behave similarly with respect to a given covariate. Note moreover that there is a total of p correspondence matrices (i.e., one per covariate). The temperature correspondence matrix visualized in Fig. 10 (a) shows a striking smoothness in the transitions between the different tasks, i.e., consecutive hours are typically modeled using the same temperature transfer functions, albeit we did not explicitely enforce this property. While all the transfer functions in Fig. 10 (b) have a similar V-shape, there are noteworthy differences. For example, TF4 compared to TF1 leads to higher load predictions for hot temperatures and lower load predictions for cold temperatures. Intuitively, we can interpret TF4 and TF1 as representing "air conditioning" and "heating" effects, respectively. This corresponds well with the hours for which these two functions are activated: TF4 during the day where most air conditioning occurs, and TF1 during the night and early morning, where most electricity is used for heating. We stress that such an analysis would have been difficult to perform with IAM. In fact, in order to detect



(a) (b)

Fig. 10. Results on the GEFCom data with $L = 4$. (a) $\mathbf{\Lambda}_{\text{temp}}$ matrix showing the activation of the temperature transfer functions for the 24 hours per day. (b) Shapes of the estimated temperature transfer functions.



(a) (b)

Fig. 11. $\mathbf{\Lambda}_{\text{temp}}$ in (a) and transfer functions in (b) when using traditional sparse coding constraint (sparsity of *global* sparse codes less than p), and OMP algorithm. In the proposed approach, we instead use a specialized and novel sparse structure constraint that does not allow more than 1 nonzero entry for each column of $\mathbf{\Lambda}_{\text{temp}}$.

the tasks that behave similarly with respect to temperature, the domain expert would have needed to carefully analyze the similarities and differences between the 24 transfer functions, and manually cluster those to infer an interpretable result. In contrast, the proposed approach provides an automatic clustering of the tasks that can be directly analyzed by the domain expert.

Finally, we emphasize once again the importance of the proposed sparsity model that imposes the sparsity of each vector $\boldsymbol{\lambda}_j^{(m)}$ to be at most one. We use a different sparse coding strategy (OMP) instead of the proposed BC-OMP that is specifically tailored for the additive model settings. The OMP algorithm is used along with the sparsity constraint $\|\boldsymbol{\lambda}^{(m)}\|_0 \leq p$ for all $m \in [N]$, where $\boldsymbol{\lambda}^{(m)}$ denote the mth column of matrix $\mathbf{\Lambda}$ (see Section III for more details on the notations). The results of this experiment are shown in Fig. 11. It can be seen that, unlike the proposed approach that yielded interpretable and smooth transitions between the different transfer functions, using a *traditional* sparse coding model does not yield such results, and rather leads to significant mixing and cancellations between the different transfer functions. In fact, the transfer functions using this sparsity model are hardly interpretable as two (or more) temperature transfer functions can be active for the same task, while our method learns disentangled factors of variation in the data and can be easily interpretable by a human expert. We finally note that similar results to those shown in Fig. 11 are obtained when using an ℓ_1 sparse coding mapping.

VII. Conclusion

In this paper, we introduced a novel multi-task learning framework for additive models with the key idea to *share transfer functions across the different tasks*. We established a connection between the proposed model and sparse dictionary learning and leveraged it to derive an efficient fitting algorithm. We further conducted a theoretical analysis of the recovery conditions of the sparse representation step; by distinguishing between coherence within and across different subdictionaries, we were able to establish recovery for a wider range of realistic settings that are particularly relevant in our multi-task learning problem. Through synthetic experiments, we showed that the proposed algorithm correctly estimates the underlying transfer functions, and outperforms competing methods in terms of predictive power. In experiments with real-world electricity demand data, we demonstrated that our proposed multi-task approach achieves competitive performance with baseline methods that learn models independently for each task, while providing models that are *more interpretable*, extracting *inherent structure* in the tasks (e.g., clustering of tasks corresponding to different customer types), and being *more robust* in settings where training data are scarce.

VIII. Acknowledgments

We thank the anonymous reviewers for their feedback that helped improve the quality of the paper. We also thank Bei Chen, Francesco Dinuzzo and Jean-Baptiste Fiot for their comments in an early stage of the paper.

Appendix

A. Proof of Theorem 2

Assume that $\mathbf{y} = \sum_{j=1}^{p} \gamma_j \mathbf{d}_{j,l_j}$. We prove by induction that the correct atoms $\mathbf{d}_{j,l_j}$ are recovered when the sufficient condition holds. Assume that, after $j \in \{0, \dots, p-1\}$ steps, BC-OMP has recovered correct atoms in the support. Therefore, it holds that the residual signal $\mathbf{r}_j \in \text{span}(\mathbf{d}_{1,l_1}, \dots, \mathbf{d}_{p,l_p})$ and we write

$$\mathbf{r}_j = \sum_{g=1}^{p} \alpha_g \mathbf{d}_{g,l_g}.$$

Since $\mathbf{y}$ is exactly p-sparse, the residual is non-zero, and $\boldsymbol{\alpha} \neq \mathbf{0}$. The atom selected by BC-OMP at step $j+1$ is optimal if and only if:

$$\max_{k \in \mathcal{A}_j} \max_{\substack{l \in [L_k] \\ l \neq l_k}} |\langle \mathbf{r}_j, \mathbf{d}_{k,l} \rangle| < \max_{k \in \mathcal{A}_j} |\langle \mathbf{r}_j, \mathbf{d}_{k,l_k} \rangle|. \tag{8}$$

We establish the recovery condition by showing a lower bound to the right hand side and an upper bound to the left hand side of Eq. (8). Note that, for any $k \in [p]$:

$$\begin{aligned} |\langle \mathbf{r}_j, \mathbf{d}_{k,l_k} \rangle| &= \left| \sum_{g=1}^{p} \alpha_g \left\langle \mathbf{d}_{g,l_g}, \mathbf{d}_{k,l_k} \right\rangle \right| \\ &= \left| \alpha_k + \sum_{g \neq k} \alpha_g \left\langle \mathbf{d}_{g,l_g}, \mathbf{d}_{k,l_k} \right\rangle \right| \\ &\geq |\alpha_k| - \sum_{g \neq k} |\alpha_g| \left| \left\langle \mathbf{d}_{g,l_g}, \mathbf{d}_{k,l_k} \right\rangle \right|. \end{aligned}$$

Moreover, by definition, $\left|\left\langle \mathbf{d}_{g,l_g}, \mathbf{d}_{k,l_k} \right\rangle\right| \leq \mu_{inter}$ for $g \neq k$. It follows that the right hand side of Eq. (8) can be bounded as follows:

$$\begin{aligned} \max_{k \in \mathcal{A}_j} |\langle \mathbf{r}_j, \mathbf{d}_{k,l_k} \rangle| &\overset{(a)}{=} \max_{k \in [p]} |\langle \mathbf{r}_j, \mathbf{d}_{k,l_k} \rangle| \\ &\geq \max_{k \in [p]} \left\{ |\alpha_k| - \mu_{inter} \sum_{g \neq k} |\alpha_g| \right\} \\ &\overset{(b)}{\geq} \|\boldsymbol{\alpha}\|_\infty - (p-1)\mu_{inter} \|\boldsymbol{\alpha}\|_\infty, \end{aligned}$$

where (a) is due to the fact that atoms $\mathbf{d}_{k,l_k}$ that are *not* in $\mathcal{A}_j$ have already been selected, and are therefore orthogonal to $\mathbf{r}_j$. Inequality (b) is obtained by bounding each term $|\alpha_g|$ by $\|\boldsymbol{\alpha}\|_\infty$.

We now exhibit an upper bound to the left hand side term of Eq. (8). We have:

$$\begin{aligned} &\max_{k \in \mathcal{A}_j} \max_{\substack{l \in [L_k]] \\ l \neq l_k}} |\langle \mathbf{r}_j, \mathbf{d}_{k,l} \rangle| \\ \leq &\max_{k \in [p]} \max_{\substack{l \in [L_k] \\ l \neq l_k}} |\langle \mathbf{r}_j, \mathbf{d}_{k,l} \rangle| \\ = &\max_{k \in [p]} \max_{\substack{l \in [L_k] \\ l \neq l_k}} \left| \alpha_k \left\langle \mathbf{d}_{k,l_k}, \mathbf{d}_{k,l} \right\rangle + \sum_{g \neq k} \alpha_g \left\langle \mathbf{d}_{g,l_g}, \mathbf{d}_{k,l} \right\rangle \right| \\ \leq &\max_{k \in [p]} |\alpha_k| \, \mu_{intra} + \mu_{inter} \sum_{g \neq k} |\alpha_g| \\ \leq &\mu_{intra} \|\boldsymbol{\alpha}\|_\infty + (p-1)\mu_{inter} \|\boldsymbol{\alpha}\|_\infty. \end{aligned}$$

Therefore, we obtain the following condition for Eq. (8) to hold:

$$\begin{aligned} \mu_{intra} \|\boldsymbol{\alpha}\|_\infty + (p-1)\mu_{inter} \|\boldsymbol{\alpha}\|_\infty \\ < \|\boldsymbol{\alpha}\|_\infty - (p-1)\mu_{inter} \|\boldsymbol{\alpha}\|_\infty. \end{aligned}$$

Since $\boldsymbol{\alpha} \neq \mathbf{0}$, we simplify the condition to:

$$\mu_{intra} + 2(p-1)\mu_{inter} < 1.$$

As the above condition holds by assumption, we conclude that Eq. (8) is satisfied and BC-OMP selects a correct atom at step $j+1$.

Once the correct support is recovered, it is straightforward to see that an orthogonal projection onto the span of the recovered atoms yields the correct coefficients. Indeed, if we have $\mathbf{y} = \sum_{j=1}^{p} \gamma'_j \mathbf{d}_{j,l_j}$, the linear independence of the atoms $\{\mathbf{d}_{j,l_j}\}$ imposes $\gamma'_j = \gamma_j$. This concludes the proof. □

B. Proof of equivalence between centered and non-centered problems

We briefly show the equivalence between the constrained optimization problem:

$$\text{(P1):} \quad \min_{\beta} \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} \sum_{t=1}^{T_j} \beta_{jt} \phi_{jt}(x_{ij}) \right)^2$$
$$\text{s.t.} \sum_{i=1}^{n} \sum_{t=1}^{T_j} \beta_{jt} \phi_{jt}(x_{ij}) = 0,$$

and the unconstrained one:

$$\text{(P2):} \quad \min_{\beta} \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} \sum_{t=1}^{T_j} \beta_{jt} s_{jt}(x_{ij}) \right)^2 ,$$

where we recall that s are the centered splines (see Section II). Note that the equivalence between both optimization problems has previously been observed and used in previous works (e.g., [22]); we provide here a brief justification for the sake of completeness.

Proposition 1. *The solutions of both optimization problems, (P1) and (P2) are equal.*

Proof. Let β_1 be an optimal solution of (P1). We evaluate the objective function of (P2) at β_1:

$$\begin{aligned}
&\sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} \sum_{t=1}^{T_j} (\beta_1)_{jt} s_{jt}(x_{ij}) \right)^2 \\
&= \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} \sum_{t=1}^{T_j} (\beta_1)_{jt} \phi_{jt}(x_{ij}) + \sum_{j=1}^{p} \sum_{t=1}^{T_j} (\beta_1)_{jt} \overline{\phi_{jt}} \right)^2 \\
&= \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} \sum_{t=1}^{T_j} (\beta_1)_{jt} \phi_{jt}(x_{ij}) + \sum_{j=1}^{p} \sum_{t=1}^{T_j} (\beta_1)_{jt} \frac{1}{n} \sum_{i=1}^{n} \phi_{jt}(x_{ij}) \right)^2 \\
&= \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} \sum_{t=1}^{T_j} (\beta_1)_{jt} \phi_{jt}(x_{ij}) \right)^2
\end{aligned}$$

We have used in the last equality that $\sum_{i=1}^{n} \sum_{t=1}^{T_j} (\beta_1)_{jt} \phi_{jt}(x_{ij}) = 0$. Therefore, the optimal value of (P2) is smaller than or equal to the optimal value of (P1).

We now show the opposite direction. Let β_2 be an optimal solution of (P2). Our goal is to show the existence of β that satisfies the centering constraints, and yields a smaller objective function than the solution of (P2). To do so, for any $j \in [p]$, we define $S_j(x) \triangleq \sum_{t=1}^{T_j} (\beta_2)_{jt} s_{jt}(x) = \sum_{t=1}^{T_j} (\beta_2)_{jt} \phi_{jt}(x) - \sum_{t=1}^{T_j} (\beta_2)_{jt} \overline{\phi_{jt}}$. Note that $S_j(x)$ is equal to a spline function minus a constant term. Since the set of splines is closed under the addition of constant terms (as the splines are the set of piecewise polynomial functions, and adding a constant does not change the degree), then $S_j(x)$ is itself a spline function, and their exists a decompostion of $S_j(x)$ into the B-spline basis as follows:

$$S_j(x) = \sum_{k} \beta_{jt} \phi_{jt}(x),$$

for some coefficents β_{jt}. It is then easy to verify that the centering constraints are satisfied for this β:

$$\begin{aligned}
&\sum_{i=1}^{n} \sum_{t=1}^{T_j} \beta_{jt} \phi_{jt}(x_{ij}) \\
&= \sum_{i=1}^{n} \sum_{t} (\beta_2)_{jt} \phi_{jt}(x_{ij}) - \sum_{i=1}^{n} \sum_{t} (\beta_2)_{jt} \overline{\phi_{jt}} \\
&= \sum_{i=1}^{n} \sum_{t} (\beta_2)_{jt} \phi_{jt}(x_{ij}) - \sum_{t} (\beta_2)_{jt} \sum_{i=1}^{n} \phi_{jt}(x_{ij}) \\
&= 0.
\end{aligned}$$

It is moreover straightforward to see, by definition, that the objective function of (P1) at β is equal to the optimal objective function of (P2), which concludes the proof. □

REFERENCES

[1] T Hastie, R Tibshirani, and J Friedman, *The elements of statistical learning*, Springer, 2009.

[2] S Wood, *Generalized additive models: an introduction with R*, CRC press, 2006.

[3] T Hastie and R Tibshirani, *Generalized additive models*, vol. 43, CRC Press, 1990.

[4] M Aharon, M Elad, and A Bruckstein, "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. on Sig. Proc.*, vol. 54, no. 11, pp. 4311–4322, 2006.

[5] I Tosic and P Frossard, "Dictionary learning," *IEEE Sig. Proc. Mag.*, vol. 28, no. 2, pp. 27–38, 2011.

[6] K Kreutz-Delgado, J Murray, B Rao, K Engan, TW Lee, and T Sejnowski, "Dictionary learning algorithms for sparse representation," *Neural computation*, vol. 15, no. 2, pp. 349–396, 2003.

[7] YC Pati, R Rezaiifar, and PS Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Asilomar Conference on Signals, Systems and Computers*, 1993.

[8] S Mallat and Z Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. on Sig. Proc.*, vol. 41, pp. 3397–3415, Dec. 1993.

[9] R Caruana, "Multitask learning," *Machine Learning*, vol. 28, pp. 41–75, 1997.

[10] B Bakker and T Heskes, "Task clustering and gating for bayesian multitask learning," *JMLR*, vol. 4, pp. 83–99, 2003.

[11] S Thrun and L Pratt, *Learning To Learn*, Kluwer Academic Publishers, November 1997.

[12] T Evgeniou, C Micchelli, and M Pontil, "Learning multiple tasks with kernel methods," in *JMLR*, 2005, pp. 615–637.

[13] A Evgeniou and M Pontil, "Multi-task feature learning," *NIPS*, vol. 19, pp. 41–48, 2007.

[14] L Jacob, J-P Vert, and F Bach, "Clustered multi-task learning: A convex formulation," in *NIPS*, 2009, pp. 745–752.

[15] H Liu, L Wasserman, and J Lafferty, "Nonparametric regression and classification with joint sparsity constraints," in *NIPS*, 2009, pp. 969–976.

[16] Z. Jiang, Z. Lin, and L. 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, 2013.

[17] J. Mairal, F. Bach, and J. Ponce, "Task-driven dictionary learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 4, pp. 791–804, 2012.

[18] S. Gao, I. Tsang, and Y. Ma, "Learning category-specific dictionary and shared dictionary for fine-grained image categorization," *IEEE Transactions on Image Processing*, vol. 23, no. 2, pp. 623–634, 2014.

[19] A. Fawzi, M. Davies, and P. Frossard, "Dictionary learning for fast classification based on soft-thresholding," *International Journal of Computer Vision*, vol. 114, no. 2-3, pp. 306–321, 2015.

[20] P. Ruvolo and E. Eaton, "Online multi-task learning via sparse dictionary optimization.," in *AAAI*, 2014, pp. 2062–2068.

[21] A. Kumar and H. Daume III, "Learning task grouping and overlap in multi-task learning," in *International Conference on Machine Learning (ICML)*, 2012.

[22] J. Huang, J. Horowitz, and F. Wei, "Variable selection in nonparametric additive models," *Annals of statistics*, vol. 38, no. 4, pp. 2282, 2010.

[23] G Davis, S Mallat, and M Avellaneda, "Adaptive greedy approximations," *Constructive approximation*, vol. 13, no. 1, pp. 57–98, 1997.

[24] C Ekanadham, D Tranchina, and E Simoncelli, "Sparse decomposition of transformation-invariant signals with continuous basis pursuit," in *IEEE ICASSP*, 2011, pp. 4060–4063.

[25] A Fawzi and P Frossard, "Classification of unions of subspaces with sparse representations," in *Asilomar Conference on Signals, Systems and Computers*, 2013, pp. 1368–1372.

[26] K Engan, S Aase, and Hakon H, "Method of optimal directions for frame design," in *IEEE ICASSP*, 1999, vol. 5, pp. 2443–2446.

[27] J Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Trans. on Inf. Theory*, vol. 50, no. 10, pp. 2231–2242, 2004.

[28] M Davenport and M Wakin, "Analysis of orthogonal matching pursuit using the restricted isometry property," *IEEE Trans. on Inf. Theory*, vol. 56, no. 9, pp. 4395–4401, 2010.

[29] L Peotta and P Vandergheynst, "Matching pursuit with block incoherent dictionaries," *IEEE Trans. on Sig. Proc.*, vol. 55, no. 9, pp. 4549–4557, 2007.

[30] Y Eldar, P Kuppinger, and H Bolcskei, "Block-sparse signals: Uncertainty relations and efficient recovery," *IEEE Trans. on Sig. Proc.*, vol. 58, no. 6, pp. 3042–3054, 2010.

[31] A Ba, M Sinn, Y Goude, and P Pompey, "Adaptive learning of smoothing functions: application to electricity load forecasting," in *NIPS*, 2012, pp. 2510–2518.

[32] S Fan and RJ Hyndman, "Short-term load forecasting based on a semi-parametric additive model," *IEEE Trans. on Power Systems*, vol. 27, no. 1, pp. 134–141, 2012.

[33] H Drucker, C Burges, L Kaufman, A Smola, V Vapnik, et al., "Support vector regression machines," *NIPS*, vol. 9, pp. 155–161, 1997.

[34] RE Fan, KW Chang, CJ Hsieh, XR Wang, and CJ Lin, "Liblinear: A library for large linear classification," *The Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

[35] CC Chang and CJ Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011.

[36] "Smart metering information paper 4: Results of electricity cost-benefit analysis, customer behavior trials and technology trials," The Commission for Energy Regulation (CER), 2011.

[37] D Fay, J Ringwood, M Condon, and M Kelly, "24-h electrical load data-a sequential or partitioned time series?," *Neurocomputing*, vol. 55, no. 3, pp. 469–498, 2003.

[38] T Hong, P Pinson, and S Fan, "Global energy forecasting competition 2012," *International Journal of Forecasting*, vol. 30, no. 2, pp. 357–363, 2014.