Structured Dimensionality Reduction for Additive Model Regression

Alhussein Fawzi Jean-Baptiste Fiot Bei Chen Mathieu Sinn Pascal Frossard

Abstract—Additive models are regression methods which model the response variable as the sum of univariate transfer functions of the input variables. Key benefits of additive models are their accuracy and interpretability on many real-world tasks. Additive models are however not adapted to problems involving a large number (e.g., hundreds) of input variables, as they are prone to overfitting in addition to losing interpretability. In this paper, we introduce a novel framework for applying additive models to a large number of input variables. The key idea is to reduce the task dimensionality by deriving a small number of new covariates obtained by linear combinations of the inputs, where the linear weights are estimated with regard to the regression problem at hand. The weights are moreover constrained to prevent overfitting and facilitate the interpretation of the derived covariates. We establish identifiability of the proposed model under mild assumptions and present an efficient approximate learning algorithm. Experiments on synthetic and real-world data demonstrate that our approach compares favorably to baseline methods in terms of accuracy, while resulting in models of lower complexity and yielding practical insights into high-dimensional real-world regression tasks. Our framework broadens the applicability of additive models to high-dimensional problems while maintaining their interpretability and potential to provide practical insights.

Index Terms—Nonparametric regression, additive models, mixed integer programming, interpretability, projection pursuit regression.

1 INTRODUCTION

With the ever increasing deployment of devices and systems for data collection, transmission and storage, real-world regression problems have become high-dimensional almost by default. A key challenge in learning high-dimensional regression models is to prevent overfitting and distinguish informative from redundant input variables. Furthermore, in many real-world applications it is paramount to learn *interpretable* models that provide domain experts with practical, easy-to-grasp insights into which are the relevant inputs, and how do they affect the outputs.

Additive models (Hastie and Tibshirani, 1990; Wood, 2006) represent the response variable as the sum of unknown *transfer functions* (also called *ridge functions*) $f_j : \mathbb{R} \to \mathbb{R}$ of the covariates: $y = \sum_{j=1}^{p} f_j(x_j) + \epsilon$. Here, y is a real-valued response variable, $\mathbf{x} = (x_1, \ldots, x_p)^T$ is a p-dimensional vector of covariates and ϵ is an error term. Additive models have been shown to yield good predictive performance on a number of real-world regression tasks, e.g., forecasting of electric load (Ba et al., 2012), air pollution (Peng and Welty, 2004), criminal incidents (Wang and Brown, 2011), etc. At the same time, the additivity assumption simplifies the structure of the models considerably and allows domain experts to grasp relations between inputs and outputs by inspecting the univariate transfer functions f_j one-by-one.

For complex, high-dimensional regression problems that involve hundreds or thousands of inputs, learning additive models with one transfer function per input variable is prone to overfitting the data and losing the model interpretability. To address these issues, feature selection and dimensionality reduction methods have been extensively studied in the literature (Su and Zhang, 2013; van der Maaten et al., 2009; Zhu et al., 2012, 2014; Guyon and Elisseeff, 2003; Zhu et al., 2013). In (Huang et al., 2010), the authors use a spline approximation for the functions f_j and introduce a group-LASSO formulation on the spline coefficients. Likewise, (Ravikumar et al., 2009) combines backfitting and LASSO for nonparametric feature selection. While these papers consider the problem of *selecting* the most relevant covariates with regard to the regression task at hand, we address in this paper the problem of *deriving* a small number r of new covariates from the p "raw" input variables ($r \ll p$). While conventional dimensionality reduction methods such as (Sparse) Principal Component Analysis (see e.g., (Jolliffe, 2005; Zou et al., 2006)) take into account only the structure of the inputs, our approach estimates the projections with regard to the regression problem at hand, i.e., it also considers the output variables.

Prior work in this direction are *additive index models* and *projection pursuit regression* (PPR) (Friedman and Stuetzle, 1981; Hastie et al., 2009) which aim at finding linear combinations of covariates as input for additive models. While providing extra flexibility, those approaches are known to suffer from their lack of interpretability (Morton, 1989) and tendency to overfitting (Zhang et al., 2008). The authors of (Zhang et al., 2008) attempt to address these issues by considering a simple sparsity prior on the linear coefficients, however, we believe that in general a more structured model is needed in order to provide both accurate and interpretable results. More recently, (Chen and Samworth, 2014) introduced shape constraints on the transfer functions, however, without considering constraints on the linear coefficients of the raw inputs.

In this paper, we introduce *Structured Dimensionality*

- *A. Fawzi and P. Frossard are with the Signal Processing Laboratory (LTS4), EPFL, Lausanne, Switzerland. E-mail: {alhussein.fawzi, pascal.frossard}@epfl.ch.*
- *J.-B. Fiot, B. Chen and M. Sinn are with IBM Research, Dublin, Ireland. E-mail: {jean-baptiste.fiot, beichen2, mathsinn}@ie.ibm.com*

Reduction for Additive Models (SDRAM), a framework for deriving covariates of additive models from high-dimensional inputs. We impose constraints which allow for the representation of structure in the input variables, prevent overfitting and facilitate the interpretation of the derived covariates and how they affect the dependent variable. In Sec. 2 we introduce our model and extend the result in (Yuan, 2011) to establish its identifiability. Sec. 3 formulates the learning algorithm and presents an efficient approximate algorithm for solving it; a key step in the derivation is the reformulation into a mixed-integer program to handle complementarity constraints (Jeroslow, 1978; Hu et al., 2008). Experiments on synthetic data and on two real-world case studies – modeling the shared bicycle system in the city of Dublin and forecasting electric load in the state of Vermont – are provided in Sec. 4. Special emphasis is put on comparing the accuracy of our approach with baseline methods and validating practical insights obtained from our model. Sec. 5 concludes the paper.

2 MODEL FORMULATION

2.1 PRELIMINARIES

We use **boldface** notations to denote vectors and matrices. For any $r \in \mathbb{N}$, we use $[r]$ to denote the set $\{1, \dots, r\}$. For any vector $\mathbf{a} = [a_1, \dots, a_n]^T \in \mathbb{R}^n$, we denote by $\text{supp}(\mathbf{a})$ the set $\{i : a_i \neq 0\}$, and use the notation $\mathbf{a}|_g$ to denote the vector $[a_{g_1}, \dots, a_{g_m}]^T$, for any $g = \{g_1, \dots, g_m\} \subseteq [n]$. For given $[n]$ and $g \subseteq [n]$, we let $\bar{g}$ denote the complement of g. We use $\|\mathbf{a}\|_p$ to denote the ℓ_p norm of $\mathbf{a}$. For any matrix $\mathbf{A} \in \mathbb{R}^{n_1 \times n_2}$, we denote by $\text{vec}(\mathbf{A})$ the vector of size $n_1 n_2$ obtained by stacking the columns of $\mathbf{A}$.

2.2 ADDITIVE INDEX MODELS

We consider the non-linear regression task

$$y_i = g(\mathbf{x}_i) + \epsilon\,,$$

for $i = 1, .., n$. Here $y_i \in \mathbb{R}$ denotes a real-valued response variable, $\mathbf{x}_i \in [-1, 1]^p$ is a normalized p-dimensional vector of covariates, g is an unknown function in $\mathbb{R}^p \to \mathbb{R}$ and ϵ_i is a white noise error term. We adopt the following regression model

$$g(\mathbf{x}) = \mu + \sum_{j=1}^{r} f_j(\mathbf{v}_j^T \mathbf{x})\,, \tag{1}$$

where $\mu \in \mathbb{R}$ is the *intercept*, $f_j : \mathbb{R} \to \mathbb{R}$ are *transfer functions* such that $f_j(0) = 0$, and $\mathbf{v}_j \in \mathbb{R}^p$ are unknown *weight vectors*. Hence, the regression model has the form of an additive model applied to the *derived* covariates $\mathbf{v}_j^T \mathbf{x}$ rather than to the "raw" input variables $\mathbf{x}$. In the literature, this class of models is known as *additive index models*. An efficient way to solve it is via the *projection pursuit regression* (PPR) algorithm (Friedman and Stuetzle, 1981), however, it has been found that without further constraints on the weight vectors $\mathbf{v}_j$, the model can be difficult to interpret (a student of one of the inventors of PPR even devoted her PhD thesis to this subject (Morton, 1989)) and tends to overfit the data – even for moderate values of r – when there is redundancy in the inputs (Zhang et al., 2008). To address these issues, we introduce a novel set of constraints on the weight vectors.

2.3 STRUCTURED DIMENSIONALITY REDUCTION

Let us formally introduce constraints **(C1)**, **(C2)** and **(C3)** on the weight vectors $\{\mathbf{v}_j\}_{j=1}^r$ in our model. Our approach features *structured dimensionality reduction* as it effectively reduces the dimensionality of the space of input variables (with regard to the regression problem at hand) while incorporating structural properties of the inputs.

(C1) **Groups.** Let $\mathcal{G} = \{g_1, \dots, g_L\}$ be a set of L pairwise disjoint subsets of $\{1, \dots, p\}$. Then,

$$\forall j \in [r], \quad \exists g \in \mathcal{G} \text{ such that } \text{supp}(\mathbf{v}_j) \subseteq g.$$

(C2) **Convex combinations.** The newly created variables are obtained from a convex combination of the input variables. That is,

$$\forall j \in [r], \quad \|\mathbf{v}_j\|_1 = 1, \quad \mathbf{v}_j \geq 0.$$

(C3) **Disjoint supports.** The input variables can take part in at most one new variable. That is,

$$\forall j, k \in [r], j \neq k, \quad \text{supp}(\mathbf{v}_j) \cap \text{supp}(\mathbf{v}_k) = \emptyset.$$

The constraint **(C1)** allows for partitioning the inputs into different user-specified groups. Each group can consist, for example, of variables of the same physical unit (e.g., degrees Celsius for temperature variables) or logical type. The derived covariates are then constrained to combine solely input variables from the same group, hence facilitating a meaningful interpretation. This constraint is crucial for the interpretation of the model, as it permits to transfer the physical meaning present in the input variables to the derived covariates. Without constraint **(C1)**, the model would be allowed to combine variables of different physical units (e.g., temperature and time), thereby creating new derived covariates that are hardly interpretable by human experts. Note that setting $\mathcal{G} = \{g\}$, with $g = \{1, \dots, p\}$ corresponds to impose no groups, as **(C1)** is then satisfied for any weight vector $\{\mathbf{v}_j\}$. We assume that the desired number of derived variables r_l for each group is given and satisfies $\sum_{l=1}^{L} r_l = r$. **(C2)** constrains the derived variables to form a convex combination of the input variables. Thus, the new variables can be seen as weighted (non-negative) averages of the inputs. This facilitates the interpretation compared to existing approaches which only impose a unit ℓ_2 norm on the weight vectors. For example, in an electric load forecasting problem with input variables representing temperature measurements from different weather stations, the constraint **(C2)** imposes derived covariates to be *spatial averages* of weather stations putting more weight on regions where demand is more sensitive to temperature. The disjoint support constraint **(C3)** prevents input variables from contributing to more than one derived covariate, thereby disentangling the different "causes" that generate the data. By assigning each input variable to at most one derived covariate (and hence one transfer function), the model becomes much easier to interpret, as the effect of the input variable on the response variable can be understood from the examination of the transfer function. Note that in models that do not satisfy constraint **(C3)**, it is very hard to track the influence of the input variables on the final response variable, as cancellations might systematically occur between the different derived covariates.

We denote by $\mathcal{V}$ the set of weight vectors that satisfy the above constraints

$$\mathcal{V} = \{\mathbf{V} = [\mathbf{v}_1|\dots|\mathbf{v}_r] \text{ such that } \{\mathbf{v}_j\}_{j=1}^r \text{ satisfy } \textbf{(C1)}, \textbf{(C2)} \text{ and } \textbf{(C3)}\}\,.$$

We consider the regression model in Eq. (1), with the additional constraint that the weight vectors $\{\mathbf{v}_j\}_{j=1}^r$ lie in $\mathcal{V}$. We call this regression model *Structured Dimensionality Reduction for Additive Models (SDRAM)*.

2.4 PRACTICAL CONSIDERATIONS

In this section, we further comment on the constraints defined in the previous section from a modeling point of view.

Choice of $\mathcal{G}$. When using the SDRAM model, the set of groups $\mathcal{G}$ is specified by the user. The set of groups is typically chosen according to physical units of the input variables. Indeed, as derived covariates are obtained as linear combinations of input variables, this allows to transfer the physical interpretation of the input variables to the derived covariates, and prevents combining two variables with different units (e.g., a speed covariate (in meters per second) and a temperature (in degrees)). In applications where $\mathcal{G}$ is not known, one can set $L = 1$, and $\mathcal{G} = \{\{1, \dots, p\}\}$, which allows to combine all input variables together.

Constraint $\|\mathbf{v}_j\|_1 = 1$. From an implementation point of view, enforcing $\|\mathbf{v}_j\|_1 = 1$, along with the normalization of the input variables in $[-1, 1]$ provides derived coordinates that are also in $[-1, 1]$, as $|\mathbf{v}_j^T\mathbf{x}| = \sum_{k=1}^p |\mathbf{v}_{jk}x_k| \leq \|\mathbf{x}\|_\infty \|\mathbf{v}_j\|_1 \leq 1$. As the feature space is stable by the feature extraction operation, the algorithm is easier to implement and avoids extrapolation issues.

Non-negativity of the weights. The convex combination constraint **(C2)** allows an interpretation of the derived covariates as average of input variables. Just like non-negative matrix factorization (Lee and Seung, 1999) yields models that are easier to inspect compared to traditional matrix factorization, we also expect the non-negativity of the weights to disallow cancellations among the input variables and thus result in more interpretable models. It should be noted moreover that the non-negativity assumption does *not* prevent having input variables that having negative correlation with the response variable, as the negative correlation can be encoded in the transfer function. Note also that the non-negativity is crucial for model identifiability, which we will present in the next section.

In applications where non-negativity is not desired, the constraint **(C2)** can be replaced by the linear constraint $\|\mathbf{v}\|_1 \leq 1$, and the algorithm we derive in the paper can still be used after applying this straightforward modification.

2.5 MODEL IDENTIFIABILITY

In this section, we establish *identifiability* of the proposed model under mild assumptions. This is an important result both from a theoretical and practical perspective; in particular, models that lack identifiability exhibit redundancy which makes it difficult to interpret them, since a model with different parameters could describe exactly the same relation between inputs and output. We first give a formal definition:

Definition 2.1 (Identifiabilty). *Assume that there exist $\{(f_j, \mathbf{v}_j)\}_{1\leq j\leq r}$ and $\{(h_j, \mathbf{w}_j)\}_{1\leq j\leq s}$ such that*

$$\forall \mathbf{x} \in \mathbb{R}^p, \quad \mu + \sum_{j=1}^r f_j(\mathbf{v}_j^T\mathbf{x}) = \nu + \sum_{j=1}^s h_j(\mathbf{w}_j^T\mathbf{x})\,, \quad (2)$$

*where $\{\mathbf{v}_j\}_{1\leq j\leq r}$ and $\{\mathbf{w}_j\}_{1\leq j\leq s}$ satisfy the constraints **(C1)**, **(C2)** and **(C3)**. Assume moreover that f_j and h_j are continuous functions, and that $f_j(0) = h_j(0) = 0$ for all j. The model is identifiable if*

1) *the intercepts agree, i.e. $\mu = \nu$,*
2) *the dimensions agree, i.e. $r = s$,*
3) *there exists a permutation $\pi\colon [r] \to [r]$ such that*

$$\forall j \in [r], \quad \begin{cases} f_j = h_{\pi(j)} \\ \mathbf{v}_j = \mathbf{w}_{\pi(j)} \end{cases}\,. \quad (3)$$

The following theorem establishes the identifiability of SDRAM.

Theorem 2.2. *Assume that there is at most one linear transfer function, then SDRAM is identifiable.*

Note that the condition of our theorem is weaker than the one for (unconstrained) additive index models (Yuan, 2011). To prove Theorem 2.2, we first show that the theorem holds whenever the transfer functions are quadratic. We then use an approach similar to (Yuan, 2011) in order to extend our result to general continuous functions. The complete proof of Theorem 2.2, together with an argument which establishes the necessity of the condition, can be found in Appendix A.

3 LEARNING ALGORITHM

In this section, we formulate the learning problem for our proposed model and derive an efficient algorithm for solving it.

3.1 FITTING PROBLEM

We consider the following learning problem

$$\min_{\substack{f_1,\dots,f_r\in\mathcal{F}\\ \mathbf{V}\in\mathcal{V}}} \quad \sum_{i=1}^n \left(y_i - \sum_{j=1}^r f_j(\mathbf{v}_j^T\mathbf{x}_i)\right)^2 + \Omega(f_1, \dots, f_r),$$

where $\mathcal{F}$ is a predefined functional space and Ω is a regularizer that operates on the transfer functions. To simplify the exposition, we assume here and in the following that the model intercept is zero. In the context of additive models, nonlinear transfer functions are commonly modeled as smoothing splines (Wood, 2006; Hastie et al., 2009; Huang et al., 2010; Ba et al., 2012), hence they take the form

$$\begin{aligned} \forall j \in [r], \quad f_j(z) &= \sum_{t=1}^k s_t(z)\beta_{jt} \\ &= \begin{bmatrix} s_1(z) & \dots & s_k(z) \end{bmatrix} \boldsymbol{\beta}_j\,, \end{aligned}$$

where $s_t\colon \mathbb{R} \to \mathbb{R}$ denotes the t-th B-spline basis function, β_{jt} its associated coefficient, and k denotes the number of spline basis functions. To simplify notation, we have dropped an extra subscript j by assuming that the same spline basis is used for all covariates. Using this representation, the B-spline coefficients $\boldsymbol{\beta}_j$ fully specify the transfer

functions. Rewriting the problem in matrix form, we obtain the following constrained least-squares problem

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^{(kr)},\mathbf{V}\in\mathcal{V}} \|\mathbf{y}-\mathbf{S}(\mathbf{V})\boldsymbol{\beta}\|_2^2+\Omega(\boldsymbol{\beta})\,,$$

with $\boldsymbol{\beta} = [\boldsymbol{\beta}_1^T,\dots,\boldsymbol{\beta}_r^T]^T$, and $\mathbf{S}(\mathbf{V}) = [\mathbf{S}_1(\mathbf{v}_1)|\dots|\mathbf{S}_r(\mathbf{v}_r)] \in \mathbb{R}^{n\times(kr)}$ where

$$\mathbf{S}_j(\mathbf{v}_j) = \begin{bmatrix} s_1(\mathbf{v}_j^T\mathbf{x}_1) & \dots & s_k(\mathbf{v}_j^T\mathbf{x}_1) \\ s_1(\mathbf{v}_j^T\mathbf{x}_2) & \dots & s_k(\mathbf{v}_j^T\mathbf{x}_2) \\ \vdots & \vdots & \vdots \\ s_1(\mathbf{v}_j^T\mathbf{x}_n) & \dots & s_k(\mathbf{v}_j^T\mathbf{x}_n) \end{bmatrix} \in \mathbb{R}^{n\times k}\,.$$

We choose the regularization function

$$\Omega(\boldsymbol{\beta}) = \Omega_{\text{ridge}}(\boldsymbol{\beta}) + \Omega_{\text{smooth}}(\boldsymbol{\beta})$$

where $\Omega_{\text{ridge}}(\boldsymbol{\beta}) = \nu\|\boldsymbol{\beta}\|_2^2$, with the parameter $\nu > 0$ determining the strength of the ridge regularizer, and

$$\Omega_{\text{smooth}}(\boldsymbol{\beta}) = \lambda\int f''(x)^2dx = \lambda\boldsymbol{\beta}^T\mathbf{C}\boldsymbol{\beta}\,,$$

with the matrix $\mathbf{C} = (\int s_i''(x)s_j''(x)dx)_{i,j}$ and $\lambda > 0$. Note that the ridge regularization term favors vectors $\boldsymbol{\beta}$ with small magnitude, while the smoothing term favors transfer functions with small second derivatives (i.e., functions that are closer to linear ones). Putting the different terms together, our learning problem is given by

$$\text{(P):}\quad \min_{\boldsymbol{\beta}\in\mathbb{R}^{(kr)},\mathbf{V}\in\mathcal{V}} \|\mathbf{y}-\mathbf{S}(\mathbf{V})\boldsymbol{\beta}\|_2^2+\lambda\boldsymbol{\beta}^T\mathbf{C}\boldsymbol{\beta}+\nu\boldsymbol{\beta}^T\boldsymbol{\beta}\,.$$

3.2 LEARNING ALGORITHM

In this section we derive an algorithm for solving the learning problem (P). From an optimization perspective, the learning problem is challenging as the weight matrix $\mathbf{V}$ is involved nonlinearly in the least-squares objective function. Moreover, the constraint $\mathbf{V}\in\mathcal{V}$ imposes new difficulties compared to the unconstrained fitting problem. We propose an alternating iterative method, where we estimate sequentially the coefficient vector $\boldsymbol{\beta}$ and the weight matrix $\mathbf{V}$. We begin by noting that, for a fixed $\mathbf{V}$, (P) reduces to a linear least squares problem that can be solved efficiently. The problem of finding $\mathbf{V}$ for a fixed coefficient vector $\boldsymbol{\beta}$, however, is much more challenging. Following a Gauss-Newton approach, we linearize the functions $s_t(\mathbf{v}_j^T\mathbf{x}_i)$ around the current estimates $\mathbf{v}_j^0$ as

$$s_t(\mathbf{v}_j^T\mathbf{x}_i) \approx s_t((\mathbf{v}_j^0)^T\mathbf{x}_i) + (\mathbf{v}_j-\mathbf{v}_j^0)^T\nabla_{\mathbf{v}}\, s_t(\mathbf{v}^T\mathbf{x}_i)\Big|_{\mathbf{v}=\mathbf{v}_j^0}\,.$$

By plugging this approximation into each entry of $\mathbf{S}(\mathbf{V})$, we obtain

$$\mathbf{S}(\mathbf{V}) \approx \mathbf{S}(\mathbf{V^0}) + \tilde{\mathbf{S}}(\mathbf{V})\,,$$

where $\tilde{\mathbf{S}}(\mathbf{V})$ is a matrix that can be written as a *linear* function of the weight vectors. Therefore, for any fixed vector $\boldsymbol{\beta}$, there exist a matrix $\mathbf{M}$ and a vector $\mathbf{b}$ not depending on $\mathbf{V}$ such that $\tilde{\mathbf{S}}(\mathbf{V})\boldsymbol{\beta} = \mathbf{b} + \mathbf{M}\text{vec}(\mathbf{V})$. The detailed derivations can be found in Appendix B. Using this approximation, the problem (P) for fixed $\boldsymbol{\beta}$ reduces to the *constrained linear* least squares problem

$$\min_{\mathbf{V}\in\mathcal{V}} \quad \|\tilde{\mathbf{y}}-\mathbf{M}\text{vec}(\mathbf{V})\|_2^2\,, \tag{4}$$

where $\tilde{\mathbf{y}} = \mathbf{y} - \mathbf{S}(\mathbf{V^0})\boldsymbol{\beta} - \mathbf{b}$. Clearly, the difficulty of the above least squares problem comes from the constraint $\mathbf{V}\in\mathcal{V}$. In order to handle condition **(C1)**, note that for any group g, the constraint $\text{supp}(\mathbf{v}_j)\subseteq g$ is equivalent to $\mathbf{v}_j|_{\overline{g}} = 0$. Therefore, as the number of derived covariates r_l belonging to group l is assumed to be known, **(C2)** is handled by imposing the constraint $\mathbf{v}_j|_{\overline{g}} = 0$ for the r_l derived covariates that belong to group g. This constraint is linear, and can be directly integrated into the optimization procedure. Similarly, the simplex constraint **(C2)** is linear in the weight vectors, thus it can be efficiently handled in the optimization. Finding an efficient formulation of the disjoint support constraint (**C3**) – known in optimization as a complementarity constraint (Jeroslow, 1978; Hu et al., 2008) – is however more challenging. We reformulate the constraint by introducing a matrix with binary entries $\mathbf{D} = \begin{bmatrix}\mathbf{d}_1 & \dots & \mathbf{d}_r\end{bmatrix}$, and obtain the following equivalent mixed-integer program formulation of the problem in Eq. (4)

$$\text{(P')}:\quad \min_{\substack{\mathbf{V}\in\mathbb{R}^{p\times r}\\ \mathbf{D}\in\{0,1\}^{p\times r}}} \quad \|\tilde{\mathbf{y}}-\mathbf{M}\text{vec}(\mathbf{V})\|_2^2$$
$$\text{subject to}\qquad \textbf{(C1)}, \textbf{(C2)} \text{ and } \begin{cases}\sum_{j=1}^r \mathbf{d}_j \le \mathbf{1}\,,\\ \forall j\in[r], \mathbf{v}_j \le \mathbf{d}_j\,,\end{cases}$$

where $\mathbf{1}$ is a vector with all entries equal to one. The introduced binary variable $\mathbf{D}$ encodes the supports of the weight vectors. The constraint $\sum_{j=1}^r \mathbf{d}_j \le \mathbf{1}$ ensures that there is at most one nonzero value in each line of $\mathbf{V}$. Note that, when $d_{jl} = 0$, the constraint $v_{jl}\le d_{jl}$ together with the positivity of the weight vectors imposes $v_{jl} = 0$. This constraint becomes redundant, however, when $d_{jl} = 1$ as the weights are upper bounded by 1 as a consequence of the simplex constraint **(C2)**.

The mixed-integer program (P') is solved using the branch-and-cut algorithm (Wolsey, 1998, Chapter 9.6), which is now efficiently implemented in many optimization toolboxes. Our learning algorithm is summarized in Algorithm 1.

Algorithm 1 SDRAM learning algorithm

1. Initialize the entries of $\mathbf{V}$ randomly using iid draws from a uniform distribution on $[0,1]$, and divide by the sum of the weights to satisfy **(C2)**.
2. For $m = 1,\dots,N$,
2.1 Update $\boldsymbol{\beta}$ by solving

$$\left(\mathbf{S}(\mathbf{V})^T\mathbf{S}(\mathbf{V}) + \lambda\mathbf{C} + \nu\mathbf{I}\right)\boldsymbol{\beta} = \mathbf{S}(\mathbf{V})^T\mathbf{y}\,.$$

2.2 Update $\mathbf{V}$ by solving the mixed-integer program (P').

4 EXPERIMENTS

In this section, we evaluate our proposed algorithm qualitatively and quantitatively on a toy example and two real-world forecasting problems.

4.1 IMPLEMENTATION AND RUNTIME

We implemented SDRAM in MATLAB and Python environments. To solve the mixed integer program, we used the MOSEK toolbox[1] in the MATLAB environment and the IBM ILOG CPLEX in the Python environment. Alternative open source toolboxes exist, e.g. GLPK[2]. On a laptop with an Intel i7 CPU, the mixed-integer program takes less than one minute to solve for all following experiments.

4.2 BASELINE METHODS AND PERFORMANCE METRICS

We compare our model to the following baseline methods:

- **Additive Models (AM)**, with the following variants: AM1 where we fit one transfer function to each input variable, and AMi for $i = 2, 3$, where we fit one transfer function to variables selected or designed using a priori knowledge of the specific problem. The regularization parameters are set via a cross-validation procedure.
- **Projection Pursuit Regression (PPR)**: We fit an unconstrained additive index model via the projection pursuit regression algorithm (Friedman and Stuetzle, 1981). We use the `ppr` function from the `stats` R-package[3].
- **Sparse Additive Models (SpAM)** (Ravikumar et al., 2009): We use the recent computationally efficient implementation of (Zhao and Liu, 2012), and set the sparsity parameter with a cross-validation procedure.
- **PCA + Additive Model (PCA+AM)**: For this two-step approach, PCA is first applied to reduce the dimension of the problem to r variables. Then, we fit an additive model on the derived variables.
- **Sparse PCA + Additive Model (SPCA+AM)**: Similar to PCA+AM, except that sparse PCA (Zou et al., 2006) is used for the dimensionality reduction step. We used the sparse PCA implementation in (Sjöstrand et al., 2012), with a sparsity value that maximizes the performance of this method.

Several metrics are used to compare the different methods:

- **Forecasting accuracy:** We compute the root mean square error, defined as

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}\,, \tag{5}$$

  where y_i and $\hat{y}_i$, for $i = 1, 2, \ldots, n$, denote the true and predicted outputs.
- **Weight matrix error:** In the experiments on synthetic data, where the true weight vectors are known, we assess the consistency of our method by considering the following metric

$$E(\mathbf{V}, \mathbf{V}^{GT}) = \frac{1}{pr}\sum_{j=1}^{r}\sum_{l=1}^{p}|v_{jl} - v_{jl}^{GT}|\,, \tag{6}$$

  where $\mathbf{V}$ and $\mathbf{V}^{GT}$ are respectively the estimated and ground truth weight matrices.

1. http://www.mosek.com/products/mosek
2. https://www.gnu.org/software/glpk/
3. http://stat.ethz.ch/R-manual/R-devel/library/stats/html/ppr.html

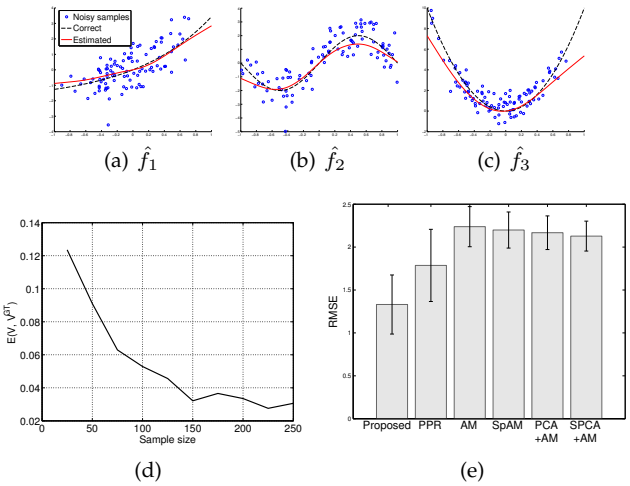


Fig. 1. (a) to (c): Transfer functions estimated using our approach (solid red line) and ground truth (dashed black line). The blue dots represent noisy samples used to train the model ($n = 100$). (d) Weight matrix error $E(\mathbf{V}, \mathbf{V}^{GT})$ as function of the sample size n. (e) RMSE between estimated signal and testing signal generated according to Eq. (7), for the different approaches. The results of experiments (d) and (e) are averaged over 300 trials.

Besides these performance measures, we also report the "complexity" of the different methods, which is measured by the number of learned functions. We finally report the sparsity of the weight matrix $\mathbf{V}$.

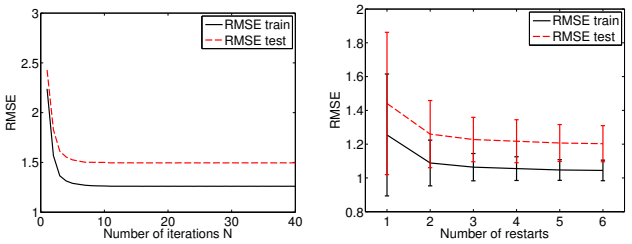
4.3 TOY EXAMPLE

In our first experiment, we generate n samples using the following additive model

$$\begin{aligned} y_i = &f_1(0.5x_{i,1} + 0.25x_{i,2} + 0.25x_{i,3}) + f_2(x_{i,4}) \\ &+ f_3(0.5x_{i,5} + 0.5x_{i,6}) + \epsilon_i\,, \end{aligned} \tag{7}$$

where $f_1(x) = 2\exp(x), f_2(x) = 2\sin(\pi x), f_3(x) = 10x^2$, the error terms ϵ_i are iid samples from a standard normal distribution, and the covariates $x_1, \ldots, x_6$ are iid samples from a uniform distribution on $[-1, 1]$. For our method, $\mathcal{G}$ is set as the trivial group $\{1, \ldots, 6\}$ (i.e. constraint **(C1)** is not used here as we allow for any combination of the $p = 6$ features) and $r = r_1 = 3$. We fix the number of iterations of our method to $N = 20$.

Figure 1 (a-c) shows the estimated transfer functions using our proposed method for a sample of size $n = 100$, together with the true transfer functions. As can be seen, our method yields good approximations of the true transfer functions, despite the relatively small sample size. We then evaluate the ability of the algorithm to estimate the true weight matrix $\mathbf{V}$. Figure 1(d) shows the metric $E(\mathbf{V}, \mathbf{V}^{GT})$ depending on the number of samples n. For low n, the error is relatively high (about the same order as the entries in $\mathbf{V}^{GT}$). As the sample size increases, the error becomes one order of magnitude lower than the entries of $\mathbf{V}^{GT}$. Finally, we evaluate the RMSE on a test set of $n = 100$ samples generated according to Eq. (7). Figure 1(e) shows that our approach yields a lower RMSE than AM (learned with 6 transfer functions, one per covariate), PPR, SpAM, as well as unsupervised dimensionality reduction techniques (PCA+AM and SPCA+AM) with 3 derived covariates. Note



(a) (b)

Fig. 2. Training and testing RMSE versus number of iterations N (left), and number of restarts (right) for the example in Sec. 4.3. The results are averaged over 50 trials.

that our approach yields a better performance than less constrained models (e.g., PPR) as the introduced constraints act as a regularizer that prevents overfitting, and significantly reduce the model complexity.

We now examine the influence of the number of iterations N on the performance of SDRAM. Figure 2(a) shows the training and testing RMSE with respect to N. After a few iterations, the algorithm reaches a stable solution. Setting $N = 20$ is therefore a conservative choice that we use in all experiments. Moreover, similarly to any nonconvex procedure, our algorithm is sensitive to initialization. To further evaluate this point, we illustrate in Fig. 2(b) the training and testing RMSE with respect to the number of restarts of SDRAM (for each restart, SDRAM is initialized randomly, and the instance yielding the lowest training RMSE is selected). It can be seen that using multiple restarts improves the performance of SDRAM on this example; we set the number of restarts to 3 in the following experiments.

4.4 SHARED BICYCLE SYSTEM DATA

In our second experiment, we consider a real-world regression problem: predicting the number of available bikes in the shared bicycle system of Dublin, Ireland. More specifically, the goal is to provide one-hour ahead forecasts of bike availability for all 44 bicycle stations across the city, using as inputs weather data, calendar information (e.g., weekday, hour of the day) and the lagged number of available bikes at all stations. A key challenge is to effectively capture correlations of bike availability across different stations and incorporate those into the predictions.

The dataset contains the number of available bikes for all 44 bike stations in the city of Dublin[4], at a sampling rate of 5 minutes, over a time period of 351 days. We use the first 200 days for training and the remaining 151 days for testing. We consider the input variables "Time of Day", "Day of Week" and "Temperature", as well as the number of available bikes at all 44 stations one hour before prediction, hence $p = 47$ in this experiment. We induce the following groups

$$\mathcal{G} = \{\{\text{"Time of Day"}\}, \{\text{"Temperature"}\}, \{\text{"Day of Week"}\}, \{\text{"Lagged availability"}\}\},$$

and use our algorithm to derive two covariates from the "Lagged availability" group (i.e., we set $r_1 = r_2 = r_3 = 1$, and $r_4 = 2$). We set the smoothing and ridge regularization parameters equal to $\lambda = \nu = 1$. Using cross-validation to optimize these parameter values is likely to improve the accuracy, but comes at extra computational costs.

We denote by AM1 and AM2 two additive models where AM1 uses all $p = 47$ input variables as covariates, while AM2 only uses 4 covariates, namely "Time of Day, "Temperature", "Day of Week" and "Lagged availability at the station to predict". In other words, AM2 ignores the number of available bikes at other stations. Table 1 provides a comparison of the different methods in terms of performance and model complexity, measured by the number of learned transfer functions. While PPR outperforms SDRAM on the training set, its performance is worse on the testing set. Confirming the findings in Zhang et al. (2008), this result suggests that, without imposing any constraints, PPR tends to overfit the data. Note that SDRAM also outperforms AM1 and AM2 on the testing set. While AM2 provides an average testing accuracy close to SDRAM, it handles the stations independently and therefore does not provide insights into correlations among different stations. Moreover, our approach compares favorably to SpAM, even if SDRAM learns much less functions. SDRAM also significantly outperforms unsupervised dimensionality reduction approaches PCA+AM, and SPCA+AM[5]. The paired Wilcoxon test shows that the improvement of SDRAM over all these methods is statistically significant at a significance level of 0.01. We also compared SDRAM with SDRAM-C2, a variant of the proposed approach where only constraint **(C2)** is active (i.e., we set $L = 1$ for **(C1)** and **(C3)** is ignored). SDRAM-C2 can be seen as a variant of Sparse PPR (Zhang et al., 2008), where the ℓ_1 norm of the weights is used as a regularizer to achieve weight sparsity. Table 1 shows that SDRAM-C2 achieves comparable accuracy with SDRAM[6]. Despite not having a direct impact on the accuracy on this task, constraints **(C1)** and **(C3)** are nevertheless crucial to obtain an interpretable model. To illustrate this point, Figure 3 displays the weight matrices obtained using SDRAM, PPR, PCA+AM and SDRAM-C2 for one particular bike station (Station 1). While competing methods yield a *dense* and *unstructured* matrix, the solution of SDRAM is *structured* and *sparse*. Quantitatively, SDRAM yields for this station a weight matrix with 83% zero entries, while the matrices obtained via PPR and PCA methods are fully dense, and SDRAM-C2 provides a matrix with 20% zeros. More importantly, PPR, PCA and SDRAM-C2 provide *unstructured weight matrices* that combine inputs of different physical types, e.g., temperature, time of day and number of available bicycles; this makes it virtually impossible to interpret the relations between inputs and outputs in a meaningful way. Conversely, our method keeps variables of different physical types separated. To highlight the interpretability of the obtained solution, Fig. 4 shows maps with the estimated weights given to the lagged input variables for the two derived covariates, along with the

4. http://www.dublinbikes.ie

5. The results for SPCA+AM are not reported in the Table 1, as the best accuracy for this experiment is reached when the sparsity parameter is equal to $p = 47$, which is equivalent to PCA+AM.

6. The difference of testing average RMSE between SDRAM and SDRAM-C2 in Table 1 is *not* statistically significant using the Wilcoxon test.

TABLE 1
Average RMSE on training and testing sets, average number of learned functions and average number of non-zero elements in $\mathbf{V}$ over 44 stations. The symbol * indicates testing RMSEs that are significantly higher than the ones obtained by SDRAM, at a significance level of 0.01.

Method		RMSE		Complexity	
		Training	Testing	# functions	$\mathbf{V}$ sparsity
SDRAM		3.21	**3.21**	5	79%
PPR		3.07	3.31*	5	0%
Additive models	AM1	3.05	3.34*	47	NA
	AM2	3.38	3.26*	**4**	NA
SpAM		3.25	3.28*	38.4	NA
PCA + AM		5.20	5.32*	5	0%
SDRAM-C2		**3.02**	**3.16**	5	20%

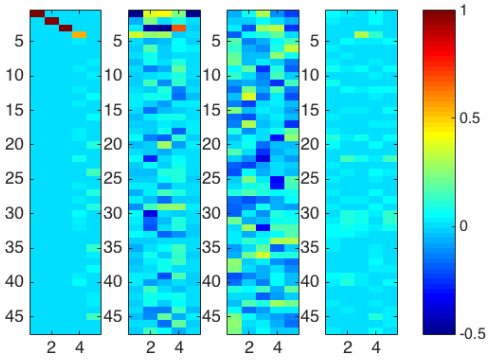


Fig. 3. Weight matrix $\mathbf{V}$ learned using SDRAM, PPR, PCA, and SDRAM-C2. The x axis denotes the derived variable number, and the y axis is the input variable number. The first three input variables are "Time of Day", "Temperature" and "Day of Week". The remaining 44 variables are the lagged variables.

associated transfer functions. In these maps, the station to predict (Station 1) is denoted with a big dot. While the first transfer function represents a *positive* correlation between the number of available bikes at time $t-1h$ and at t, the second transfer function shows a *negative* correlation. Interestingly, one can see that the first derived variable essentially corresponds to the lagged number of available bikes at the station to predict (Station 1). On the other hand, the second derived variable combines several stations that are negatively correlated with the response variable. Note that this intuitive separation of the covariates is essentially due to the disjoint support constraint **(C3)** which allows to disentangle positive and negatively correlated stations. To further show this point, Fig. 5 shows the estimated maps when constraint **(C3)** *is not active*. Unlike in Fig. 4, some stations are active in both maps (e.g., see the station represented with a big dot that is maximally active in both cases). This results in entangled positive and negative correlation effects for each station, which leads to a very difficult interpretation as cancellations systematically occur between the different transfer functions. On the other hand the obtained maps in Fig. 4 can be readily interpreted: the bike station for which the predictions are computed lies in the commercial

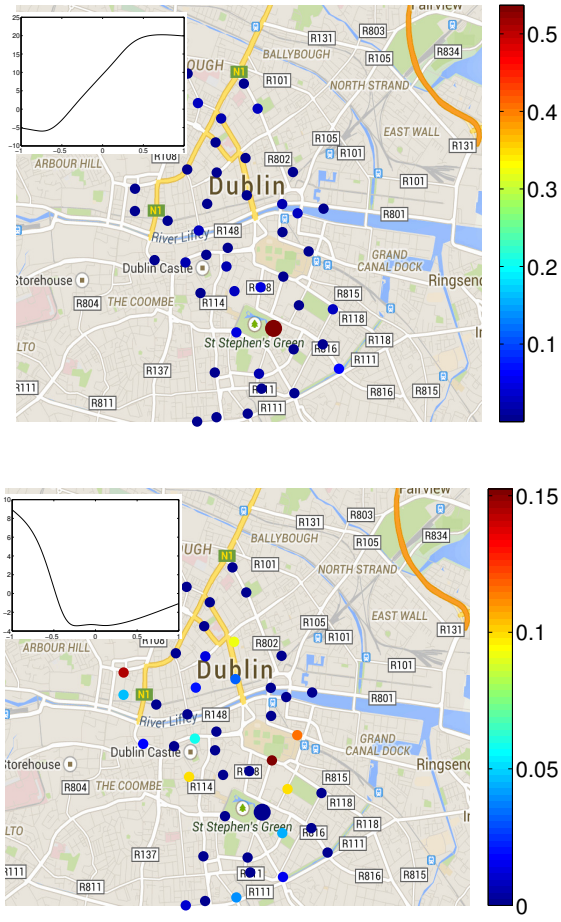


Fig. 4. Public bike availability forecasting: weights learned with SDRAM for the first and second derived covariate, shown on a map of Dublin with the 44 stations. The big dot denotes the station where prediction occurs. The shape of the corresponding transfer function is shown in the top left corner of each map.

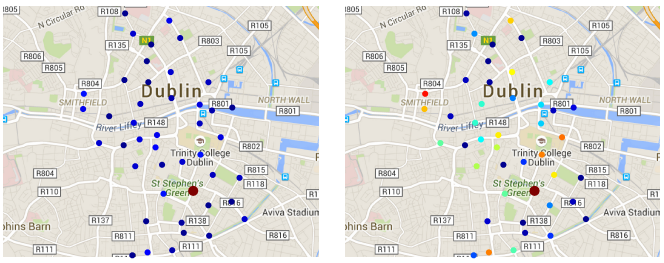


Fig. 5. Weights learned using our method *without imposing constraint* C3. Note that some stations are active in both maps. For example, the station where the prediction occurs (represented by a big dot) is the maximally active station in both maps.

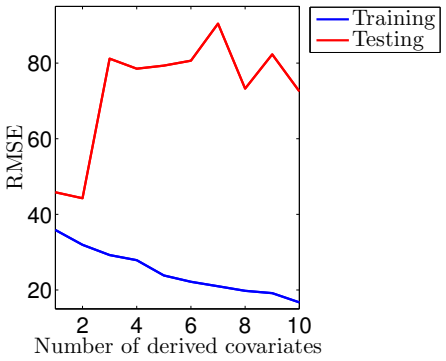


Fig. 6. RMSE of the PPR method as a function of the number of derived covariates.

heart of the city and close to important transportation hubs. The negative correlation is due to the mobility patterns of Dublin commuters: the three top weighted stations are Smithfield North, Pearse and Leinster Street. The first one is located in a residential area and the latter two are on a university campus. In mornings and evenings, people commute by bike from their homes in the residential area to their working places in the city center. In addition, students pick up bikes at this tranportation hub to complete the last mile of their journey to the university campus.

4.5 ELECTRIC LOAD FORECASTING

In our last experiment, we apply our algorithm to short-term electric load forecasting. Note that additive models have been quite successfully applied to this task previously, with covariates including calendar information, weather data as well as auto-regressive and lagged features (Fan and Hyndman, 2012). A difficult problem is how to optimally incorporate localized weather measurements, i.e., how to weight the input from weather stations in different regions in order to predict electric load at the state level. The authors of (Goude et al., 2014) state this as an open problem and explicitely mention the need for automatic covariate selection methods. In (Ba et al., 2012), weather stations are weighted according to the relative load in that particular region. Similarly, one could consider socio-economic indicators (population density, type of heating in different parts of the state, etc), however this information is not always available. Our solution is to simulateously learn the weights and transfer functions from the data.

The dataset comes from two sources: hourly electric load data for the state of Vermont, USA, from ISO New England[7] and temperature data from 40 weather stations from MADIS[8]. The prediction task is to forecast electrical loads 24 hours ahead of time. The input variables in our model are "Time of Year", "Time of Day", "Day of Week", "Lag load" and "T", i.e., the temperatures from the 40 weather stations. Similarly to the model in (Fan and Hyndman, 2012), we also consider "T_{lag24}" (the temperatures from the 40 weather stations lagged by 24 hours), "T_{mean24}", "T_{min24}", "T_{max24}" (the mean, minimum and maximum over the past 24 hours for each station) and "T_{mean7}" (the mean over the past seven days for each station). We enforce the following groups in the derivation of the covariates

$$\begin{aligned}\mathcal{G} =&\{\{\text{"Time of Year"}\},\{\text{"Time of Day"}\},\\ &\{\text{"Day of Week"}\},\{\text{"Lag load"}\},\{T\},\\ &\{T_{lag24}\},\{T_{mean24}\},\{T_{min24}\},\\ &\{T_{max24}\},\{T_{mean7}\}\},\end{aligned}$$

and derive one variable per group in our method (i.e., we set $r_l = 1$ for $l \in [10]$). The dataset is split as follows: omitting any time points for which we have missing load or temperature values, we use 9013 observations between 4 January, 2011 and 31 December, 2012 for training, and 4149 observations between 1 January, 2013 and 31 January, 2014 for testing.

We denote by AM1-3 the three additive models defined as follows. AM1 learns one transfer function for each of the 244 input variables. AM2 and AM3 learn one transfer function for each of the 10 groups, where AM2 uses the average of the 40 temperature inputs as covariates, and AM3 selects the inputs from the city of Burlington, which is the area with the highest population density in Vermont.

Table 2 shows that SDRAM provides the best performance: it has the lowest testing RMSE, a limited number of transfer functions, and provides a sparse dimensionality reduction matrix. The second lowest testing RMSE is obtained by SpAM. However, 1) SpAM learns approximately 8 times more functions than than SDRAM and 2) SpAM acts as a feature selection algorithm, and does not derive covariates out of existing input variables. Note also that the proposed method compares favorably to SDRAM-C2, which only considers constraint **(C2)**. It should be noted moreover that SDRAM-C2 systematically yields derived covariates that combine input variables with different physical units, leading to a loss of interpretability. Moreover, input variables take part in many derived covariates as **(C3)** is not active, which makes it difficult to track the effect of input variables on the response variable. AM1 and PPR methods suffer from overfitting as they provide good training accuracy but do not generalize well on the test set. To further study this behaviour, we evaluated the testing accuracy of PPR as a function of the number of derived covariates (see Fig. 6). We have observed that PPR strongly overfits the data when $r \geq 3$, leading to very poor testing accuracy. Hence,

7. http://www.iso-ne.com/isoexpress/web/reports/load-and-demand/-/tree/dmnd
8. http://madis.noaa.gov/

TABLE 2
Model accuracy (RMSE) and complexity on the electric load forecasting problem.

Method		RMSE		Complexity	
		Training	Testing	# functions	$\mathbf{V}$ sparsity
SDRAM		26.6	**27.6**	10	**98%**
PPR	$r = 1$	35.8	45.8	**1**	0%
	$r = 2$	31.9	44.2	2	0%
	$r = 10$	**16.7**	72.5	10	0%
Additive models	AM1	24.4	28.3	244	NA
	AM2	28.1	28.8	10	NA
	AM3	27.9	28.5	10	NA
SpAM		26.0	28.1	78	NA
PCA + AM	$r = 10$	39.2	39.7	10	0%
	$r = 100$	34.6	40.8	100	0%
SPCA + AM	$r = 10$	28.7	31.2	10	96%
	$r = 100$	27.1	29.6	100	96%
SDRAM-C2		26.3	28.3	10	9.1%

the constraints on $\mathbf{V}$ are crucial to avoid overfitting. As for PCA + AM and SPCA + AM, it can be noted that these unsupervised dimensionality reduction approaches provide significantly lower accuracy than SDRAM.

To further study the interpretability of the obtained solution, Fig. 7 shows the maps of the weights associated with the different weather stations in the derivation of the temperature-related covariates. Interestingly, for most derived covariates, our algorithm selects stations in the Burlington area, which has the highest population density in Vermont. Moreover, there is also a representative selection of stations in the Western/Eastern part of Vermont, which have warmer/colder climate, respectively[9].

5 CONCLUSION

We proposed a novel framework for learning additive models with a moderate number of covariates derived from a potentially large set of input variables. Our approach allows for the representation of structure in the input variables, which helps to prevent overfitting and leads to models that provide practical insights into relations between inputs and output. We established identifiability of the proposed model under mild assumptions on the transfer functions. We derived an efficient learning algorithm that alternates between a regularized least squares problem and a mixed-integer problem. We conducted experiments on synthetic and real-world data; the results showed that SDRAM outperforms baseline methods and highlighted the importance of the proposed contraints. Our work significantly broadens the applicability of additive models to high-dimensional problems while maintaining their interpretability and potential to provide practical insights.

9. http://www.nws.noaa.gov/climate/local_data.php?wfo=BTV, see Vermont Annual Mean High/Low Temperature

Acknowledgments. The authors would like to thank the associate editor and the anonymous reviewers for their valuable comments and references that helped to improve the quality of this paper.

APPENDIX A
PROOF OF THEOREM 2.2

A.1 Preliminary results

Our proof relies on a number of results that we give in this section. To start with, the following result establishes identifiability when we have only one ridge function.

Proposition A.1. *Suppose that f and h are functions (not identically zero) such that $f(0) = h(0) = 0$. Assume that*

$$\forall \mathbf{x} \in \mathbb{R}^p, \quad f(\mathbf{v}^T\mathbf{x}) = h(\mathbf{w}^T\mathbf{x}), \tag{8}$$

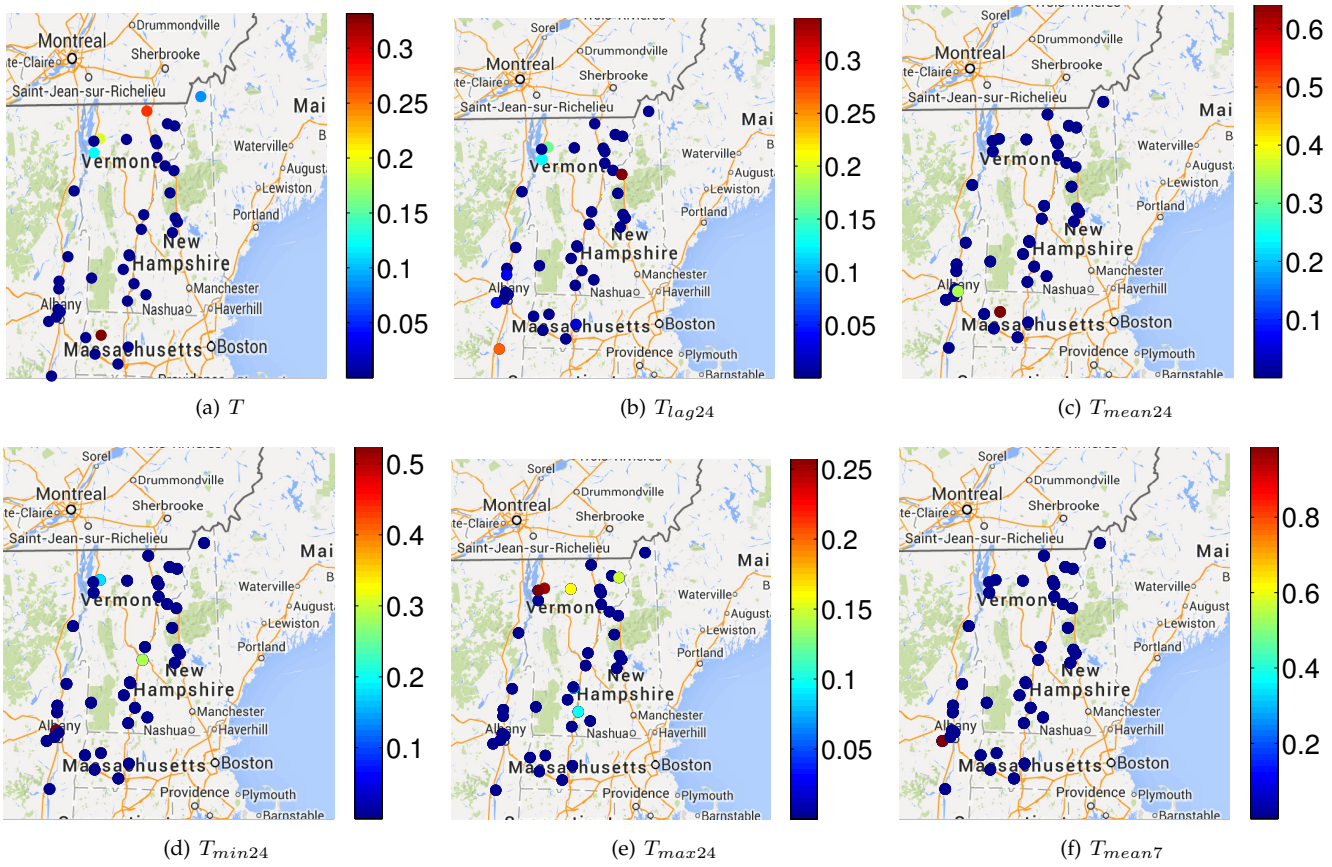
with $\mathbf{v}$ and $\mathbf{w}$ non-negative vectors with unit ℓ_1 norm. Then $\mathbf{v} = \mathbf{w}$, and $f = h$.

Proof. We proceed by contradiction, and assume that $\mathbf{v}$ and $\mathbf{w}$ are not collinear. In other words, assume that $\text{span}(\mathbf{v}) \neq \text{span}(\mathbf{w})$, which is equivalent to $\text{span}(\mathbf{v})^\perp \neq \text{span}(\mathbf{w})^\perp$. Moreover, $\text{span}(\mathbf{v})^\perp$ is not strictly included in $\text{span}(\mathbf{w})^\perp$, as both subspaces have the same dimension. Therefore, there exists $\mathbf{x}_0$ such that $\mathbf{x}_0 \perp \mathbf{v}$ and $\mathbf{x}_0 \notin \text{span}(\mathbf{w})^\perp$. In other words, $\mathbf{v}^T\mathbf{x}_0 = 0$, and $\mathbf{w}^T\mathbf{x}_0 \neq 0$. For any $\mu \in \mathbb{R}$, we therefore have

$$\begin{aligned} f(\mathbf{v}^T(\mu\mathbf{x}_0)) = f(\mu\mathbf{v}^T\mathbf{x}_0) = f(0) = 0 = h(\mathbf{w}^T(\mu\mathbf{x}_0)) \\ = h(\mu \underbrace{\mathbf{w}^T\mathbf{x}_0}_{\neq 0}). \end{aligned} \tag{9}$$

We therefore obtain $h(z) = 0$ for all z, which contradicts our assumption. Hence, we conclude that $\mathbf{v} = \lambda\mathbf{w}$ for some non-negative real value λ. Since $\|\mathbf{v}\|_1 = \|\mathbf{w}\|_1$, we therefore get $\lambda = 1$, and $\mathbf{v} = \mathbf{w}$. Hence, $f = h$. □

Using this result, we establish identifiability when ridge functions are quadratic



(a) T (b) T_{lag24} (c) T_{mean24}

(d) T_{min24} (e) T_{max24} (f) T_{mean7}

Fig. 7. Electric load forecasting: weights learned by SDRAM for the various temperature-based covariates, shown on a map of Vermont with the 40 temperature stations.

Proposition A.2. *Let $\{\mathbf{v}_j\}_{j=1}^r$ and $\{\mathbf{w}_j\}_{j=1}^s$ be weight vectors that satisfy constraints **(C1)**, **(C2)** and **(C3)**. Suppose that*

$$\forall \mathbf{x} \in \mathbb{R}^p, \quad \sum_{j=1}^{r} f_j(\mathbf{v}_j^T \mathbf{x}) = \sum_{j=1}^{s} h_j(\mathbf{w}_j^T \mathbf{x}) \,, \tag{10}$$

where $\{f_j\}_{j=1}^r$ and $\{h_j\}_{j=1}^s$ are quadratic functions with at most one linear function. We assume moreover that the functions are not identically zero and satisfy $f_j(0) = h_j(0) = 0$. Then, $r = s$ and there exists a permutation π such that, for all $j \in [r]$

$$\mathbf{v}_j = \mathbf{w}_{\pi(j)}, \quad f_j = h_{\pi(j)} \,. \tag{11}$$

Proof. Notice first that in order to prove identifiability in this case, it is sufficient to prove the following statement

$$\forall j \in [r], \quad \exists \pi(j) \text{ such that } \operatorname{supp}(\mathbf{v}_j) = \operatorname{supp}(\mathbf{w}_{\pi(j)}) \,. \tag{12}$$

Indeed, if Eq. (12) holds, then by evaluating Eq. (10) at $\mathbf{x}$ such that $\operatorname{supp}(\mathbf{x}) = \operatorname{supp}(\mathbf{v}_j)$, we get

$$f_j(\mathbf{v}_j^T \mathbf{x}) = h_{\pi(j)}(\mathbf{w}_{\pi(j)}^T \mathbf{x}) \,,$$

where we used the disjoint supports assumption and the fact that $f_j(0) = h_k(0) = 0$ for all j, k. The above equality generalizes to any $\mathbf{x}$ in $\mathbb{R}^p$ as $\mathbf{v}_j$ and $\mathbf{w}_{\pi(j)}$ have zero entries outside of $\operatorname{supp}(\mathbf{v}_j)$

$$\forall \mathbf{x} \in \mathbb{R}^p, \quad f_j(\mathbf{v}_j^T \mathbf{x}) = h_{\pi(j)}(\mathbf{w}_{\pi(j)}^T \mathbf{x}) \,. \tag{13}$$

We therefore obtain from Proposition A.1 that $\mathbf{v}_j = \mathbf{w}_{\pi(j)}$ and $f_j = h_{\pi(j)}$. Note moreover that π is one-to-one as $\pi(j_1) = \pi(j_2)$ would imply $\operatorname{supp}(\mathbf{v}_{j_1}) = \operatorname{supp}(\mathbf{v}_{j_2})$ which contradicts the disjoint support assumption. We therefore get $r = s$.

We now focus on proving Eq. (12). To do that, assuming the functions are quadratic, the main idea is to look at the monomials of degree 2 (i.e., of the form $x_a x_b$) in Eq. (10). The equality of the monomials in Eq. (10) imposes $\bigcup_j \operatorname{supp}(\mathbf{v}_j) \times \operatorname{supp}(\mathbf{v}_j) = \bigcup_j \operatorname{supp}(\mathbf{w}_j) \times \operatorname{supp}(\mathbf{w}_j)$, from which we can see that the supports of $\mathbf{v}_j$ and $\mathbf{w}_j$ have to be the same (up to a permutation), due to the disjoint support constraint. More formally, let us proceed by contradiction and assume that Eq. (12) does not hold. There exists j_0 for which

$$\forall j \in [s], \quad \operatorname{supp}(\mathbf{v}_{j_0}) \neq \operatorname{supp}(\mathbf{w}_j) \,. \tag{14}$$

Then,

$$\left.\begin{array}{l} \forall \mathbf{x} \in \mathbb{R}^p \\ \operatorname{supp}(\mathbf{x}) = \operatorname{supp}(\mathbf{v}_{j_0}) \end{array}\right\}, \quad f_{j_0}(\mathbf{v}_{j_0}^T \mathbf{x}) = \sum_{j=1}^{s} h_j(\mathbf{w}_j^T \mathbf{x}) \,. \tag{15}$$

We first examine the case where f_{j_0} is linear. If this holds, then the right hand side of Eq. (15) also has to be linear. Since there is at most one linear function, the above equality becomes $a\mathbf{v}_{j_0}^T \mathbf{x} = b\mathbf{w}_{j_1}^T \mathbf{x}$ for all $\mathbf{x}$ with the same support as $\mathbf{v}_{j_0}$, for some a, b, and index j_1. If $\operatorname{supp}(\mathbf{v}_{j_0}) \not\subset \operatorname{supp}(\mathbf{w}_{j_1})$, then there exists k such that $v_{j_0 k} \neq 0$ and $w_{j_1 k} = 0$. By setting $\mathbf{x} = \mathbf{e}_k$, we get $a v_{j_0 k} = 0$, and therefore $a = 0$. Since the functions are not identically zero, this cannot hold and we have $\operatorname{supp}(\mathbf{v}_{j_0}) \subset \operatorname{supp}(\mathbf{w}_{j_1})$. In that case, we have $b\mathbf{w}_{j_1}^T \mathbf{x} = a\mathbf{v}_{j_0}^T \mathbf{x}$ for all $\mathbf{x}$ such that $\operatorname{supp}(\mathbf{x}) = \operatorname{supp}(\mathbf{w}_{j_1})$, and we obtain $b = 0$ for the same reasons above. Therefore, f_{j_0} cannot be linear.

Let us now examine the case where f_{j_0} is a quadratic (non-linear) function. Assume first that $\operatorname{supp}(\mathbf{v}_{j_0}) \not\subset \operatorname{supp}(\mathbf{w}_j)$ for all j. If Eq. (15) is to hold, there exists at least one j such that $\operatorname{supp}(\mathbf{v}_{j_0}) \cap \operatorname{supp}(\mathbf{w}_j) \neq \emptyset$, and let j_1 be such an index. Denote by k an element in $\operatorname{supp}(\mathbf{v}_{j_0}) \cap \operatorname{supp}(\mathbf{w}_{j_1})$. Since $\operatorname{supp}(\mathbf{v}_{j_0}) \not\subset$

$\text{supp}(\mathbf{w}_{j_1})$ there exists an element $l \in \text{supp}(\mathbf{v}_{j_0})$ and not in $\text{supp}(\mathbf{w}_{j_1})$. Therefore, the cross-term $x_k x_l$ belongs to the left hand side of Eq. (15), but not to the right hand side. This cannot hold, and we conclude that there exists an element j_1 such that $\text{supp}(\mathbf{v}_{j_0}) \subset \text{supp}(\mathbf{w}_{j_1})$. Note that we have $h_{j_1}(\mathbf{w}_{j_1}^T \mathbf{x}) = \sum_j f_j(\mathbf{v}_j^T \mathbf{x})$ for all $\mathbf{x}$ such that $\text{supp}(\mathbf{x}) = \text{supp}(\mathbf{w}_{j_1})$. As before, there exists an element $k \in \text{supp}(\mathbf{v}_{j_0}) \cap \text{supp}(\mathbf{w}_{j_1})$, and $l \in \text{supp}(\mathbf{w}_{j_1})$, but $l \notin \text{supp}(\mathbf{v}_{j_0})$. Therefore, the previous equality has the cross-term $x_k x_l$ on the left hand side, but not on the right hand side. This concludes the proof of the proposition. □

In order to extend our proof from quadratic ridge functions to general continuous functions in Section A.2, we rely on the following result from Khatri and Rao (1968).

Lemma A.3. *Consider the functional equation*

$$\phi_1(\boldsymbol{\alpha}_1^T \mathbf{t}) + \cdots + \phi_r(\boldsymbol{\alpha}_r^T \mathbf{t}) = \xi_1(t_1) + \cdots + \xi_p(t_p)$$

defined for $|t_i| \leq \delta$, $i = 1, \ldots, p$, where $\delta > 0$, $\mathbf{t}$ represents the column vector of variables $t_1, \ldots t_p$, and $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_r$ are the column vectors of a $p \times r$ matrix $\mathbf{A}$. Let $\mathbf{A}$ be of full column rank such that each column has at least two non-zero entries. Then, $\phi_1, \ldots, \phi_r$ and $\xi_1, \ldots, \xi_p$ are all quadratic functions.

A.2 Proof of Theorem 2.2

Let us note $\mathcal{F} \triangleq \{f \colon \mathbb{R} \to \mathbb{R} \ ; \ f \text{ continuous}, f(0) = 0 \text{ and } f \text{ is not identically zero}\}$. Let us assume $\{(f_j, \mathbf{v}_j) \in \mathcal{F} \times \mathbb{R}^p\}_{1 \leq j \leq r}$ and $\{(h_j, \mathbf{w}_j) \in \mathcal{F} \times \mathbb{R}^p\}_{1 \leq j \leq s}$ are such that

$$(H_r) : \begin{cases} r, s \leq p\,, \\ \forall \mathbf{x} \in \mathbb{R}^p, \quad \mu + \sum_{j=1}^r f_j(\mathbf{v}_j^T \mathbf{x}) = \nu + \sum_{j=1}^s h_j(\mathbf{w}_j^T \mathbf{x})\,, \\ \{\mathbf{v}_j\}_{1 \leq j \leq r} \text{ satisfy the constraints } \textbf{(C1)}, \textbf{(C2)} \text{ and } \textbf{(C3)}\,, \\ \{\mathbf{w}_j\}_{1 \leq j \leq s} \text{ satisfy the constraints } \textbf{(C1)}, \textbf{(C2)} \text{ and } \textbf{(C3)}\,, \\ \text{At most one } f_j \text{ (and one } h_j) \text{ is linear}\,. \end{cases}$$

First, using $\mathbf{x} = 0$ gives $\mu = \nu$.

Without loss of generality, we also assume $r \leq s$. We proceed by induction on r to show that the following property

$$(P_r) : \qquad (H_r) \text{ implies identifiability}$$

holds for all r.

A.2.0.1 Initialization ($r = 1$): First we complete the set of orthogonal vectors $\{\mathbf{w}_j\}_{1 \leq j \leq s}$ in an orthogonal basis of $\mathbb{R}^p$ and note $\mathbf{W} \triangleq (\mathbf{w}_1, \ldots, \mathbf{w}_p)$. We also define $\mathbf{u}_j \triangleq \mathbf{W}^{-1} \mathbf{v}_j$ for $1 \leq j \leq r$. We have

$$\forall \mathbf{x} \in \mathbb{R}^p, \quad f_1(\mathbf{v}_1^T \mathbf{x}) = \sum_{j=1}^s h_j(\mathbf{w}_j^T \mathbf{x})\,, \tag{16}$$

$$\forall \mathbf{x} \in \mathbb{R}^p, \quad f_1(\mathbf{u}_1^T \mathbf{W}^T \mathbf{x}) = \sum_{j=1}^s h_j(\mathbf{e}_j^T \mathbf{W}^T \mathbf{x})\,, \tag{17}$$

where $\mathbf{e}_j$ is the vector with all zero elements, except the jth element is equal to one.

Now we do the change of variable $\mathbf{z} = \mathbf{W}^T \mathbf{x}$ and obtain

$$\forall \mathbf{z} \in \mathbb{R}^p, \quad f_1(\mathbf{u}_1^T \mathbf{z}) = \sum_{j=1}^s h_j(z_j)\,. \tag{18}$$

Setting $\mathbf{z} = z_k \mathbf{e}_k$, we get

$$\text{For } 1 \leq k \leq s : \quad \forall z_k \in \mathbb{R}, \quad f_1(u_{1k} z_k) = h_k(z_k)\,, \tag{19}$$

$$\text{For } s < k \leq p : \quad \forall z_k \in \mathbb{R}, \quad f_1(u_{1k} z_k) = 0\,. \tag{20}$$

Note that if $u_{1k} = 0$ for some $1 \leq k \leq s$, then using equation (19) would imply that $h_k = 0$, which is impossible since $h_k \in \mathcal{F}$. Also, if $u_{1k} \neq 0$ for some $s < k \leq p$, then using equation (20) would imply $f_1 = 0$, which is impossible since $f_1 \in \mathcal{F}$. So we have $\mathbf{u}_1 = (u_{11}, ..., u_{1s}, 0, \ldots, 0)$ with $u_{11}, \ldots, u_{1s} \neq 0$. Assuming $s > 1$, we take $\mathbf{z} = (z_1, z_2, 0, \ldots, 0)$ and using (18) and (19) we get

$$\begin{aligned} \forall z_1, z_2 \in \mathbb{R}, \quad f_1(u_{11} z_1 + u_{12} z_2) &= h_1(z_1) + h_2(z_2) \\ &= f_1(u_{11} z_1) + f_1(u_{12} z_2)\,. \end{aligned} \tag{21}$$

Therefore f_1 satisfies Cauchy's functional equation, so it is $\mathbb{Q}$-linear. Since it is also continuous, f_1 is ($\mathbb{R}$-)linear, and by (19) so are h_1 and h_2. This is impossible, so $s = 1 = r$. Therefore we have $\mathbf{u}_1 = \lambda \mathbf{e}_1$ and (18) becomes

$$\forall z_1 \in \mathbb{R}, \quad f_1(\lambda z_1) = h_1(z_1)\,. \tag{22}$$

We also get $\mathbf{v}_1 = \mathbf{W}\mathbf{u}_1 = \lambda \mathbf{W}\mathbf{e}_1 = \lambda \mathbf{w}_1$. Since $\|\mathbf{v}_1\|_1 = \|\mathbf{w}_1\|_1 = 1$, we get $\lambda = \pm 1$. The positivity of $\mathbf{v}_1$ and $\mathbf{w}_1$ gives $\lambda = 1$, and thus $\mathbf{v}_1 = \mathbf{w}_1$ and $f_1 = h_1$.

A.2.0.2 Induction: Now let us assume the hypotheses (H_{r+1}) and (P_r) hold. The strategy is to show that $f_{r+1} = h_s$ and $\mathbf{v}_{r+1} = \mathbf{w}_s$ (up to a permutation of the terms), and (H_r) holds. Calling (P_r) would then terminate the proof of (P_{r+1}).

The same change of variable as in the initialization gives

$$\forall \mathbf{z} \in \mathbb{R}^p, \quad \sum_{j=1}^{r+1} f_j(\mathbf{u}_j^T \mathbf{z}) = \sum_{j=1}^s h_j(z_j)\,. \tag{23}$$

First case: all $\{\mathbf{u}_j\}_{1 \leq j \leq r+1}$ have at least two non-zero entries. Then, using Lemma A.3, all ridge functions are quadratic. If the ridge functions contain at most one linear function, our model is identifiable according to Proposition A.2. Otherwise, the assumption is not satisfied.

Second case: there exists one $\mathbf{u}_j$ with one non-zero entry. Without loss of generality, let us say this $\mathbf{u}_j$ is $\mathbf{u}_{r+1}$, and with a permutation of coordinates take $\mathbf{u}_{r+1} = \lambda \mathbf{e}_s$, for some λ. Using the equality $\mathbf{v}_{r+1} = \mathbf{W}\mathbf{u}_{r+1} = \lambda \mathbf{w}_s$ and the unit norm constraints on $\mathbf{v}_{r+1}, \mathbf{w}_s$ we get $\lambda = 1$. We therefore have $\mathbf{u}_{r+1} = \mathbf{e}_s$, and $\mathbf{v}_{r+1} = \mathbf{w}_s$. Note also that

$$\begin{aligned} \forall j \in [r], \quad \mathbf{v}_j^T \mathbf{v}_{r+1} = \mathbf{v}_j^T \mathbf{w}_s &= (\mathbf{W}\mathbf{u}_j)^T \mathbf{w}_s \\ &= \mathbf{u}_j^T \mathbf{e}_s \|\mathbf{w}_s\|_2^2 \\ &= u_{js} \|\mathbf{w}_s\|_2^2\,, \end{aligned} \tag{24}$$

where we have used the fact that the columns of $\mathbf{W}$ are orthogonal. Since the $\mathbf{v}_j$s are orthogonal to each other, we have $\mathbf{v}_j^T \mathbf{v}_{r+1} = 0$, and therefore $u_{js} = 0$ for all $j \in [r]$ as $\mathbf{w}_s$ is a nonzero vector. By rewriting Eq. (23) and setting $z_s = 0$, we have

$$\left.\begin{matrix} \forall \mathbf{z} \in \mathbb{R}^p \\ z_s = 0 \end{matrix}\right\}, \quad \sum_{j=1}^r f_j(\mathbf{u}_j^T \mathbf{z}) = \sum_{j=1}^{s-1} h_j(z_j)\,, \tag{25}$$

Moreover, since $u_{js} = 0$ for all $j \in [r]$, the above equality is valid for all $\mathbf{z} \in \mathbb{R}^p$, and we have

$$\forall \mathbf{z} \in \mathbb{R}^p, \quad \sum_{j=1}^r f_j(\mathbf{u}_j^T \mathbf{z}) = \sum_{j=1}^{s-1} h_j(z_j)\,. \tag{26}$$

Using the change of variables, we therefore get

$$\forall \mathbf{x} \in \mathbb{R}^p, \quad \sum_{j=1}^r f_j\left(\mathbf{v}_j^T \mathbf{x}\right) = \sum_{j=1}^{s-1} h_j(\mathbf{w}_j^T \mathbf{x})\,. \tag{27}$$

which corresponds to (H_r). By calling (P_r), we have $r = s - 1$ and there exists a permutation $\pi : [r] \to [r]$ such that

$$\forall j \in [r], \quad f_j = h_{\pi(j)}, \quad \mathbf{v}_j = \mathbf{w}_{\pi(j)}\,. \tag{28}$$

We therefore have $f_{r+1}(\mathbf{v}_{r+1}^T \mathbf{x}) = h_s(\mathbf{w}_s^T \mathbf{x})$, and using the equality $\mathbf{v}_{r+1} = \mathbf{w}_s$, we conclude that $f_{r+1} = h_s$. (P_{r+1}) therefore holds.

A.3 Tightness of the condition in Theorem 2.2

We give the following counter-example to show that the assumption requiring at most one linear transfer function is necessary

$$\forall \mathbf{x} \in \mathbb{R}^3, \quad \frac{1}{2}\left(\begin{bmatrix}1 & 0 & 0\end{bmatrix}\mathbf{x}\right) + \left(\begin{bmatrix}0 & \frac{1}{2} & \frac{1}{2}\end{bmatrix}\mathbf{x}\right) = \left(\begin{bmatrix}\frac{1}{2} & \frac{1}{2} & 0\end{bmatrix}\mathbf{x}\right) + \frac{1}{2}\left(\begin{bmatrix}0 & 0 & 1\end{bmatrix}\mathbf{x}\right). \tag{29}$$

The above model is unidentifiable as the left-hand side and right hand side models are equal for all $\mathbf{x} \in \mathbb{R}^3$, yet the model parameters are different. Note also that the weight vectors in the above example are admissible as they satisfy the constraints of our model.

We finally highlight the fact that, unlike the general (unconstrained) PPR model where identifiability does not hold for quadratic transfer functions Yuan (2011), the proposed (constrained) model is identifiable in that case.

Appendix B
Derivation of the objective function linearization

In this section, we give the derivations used in our optimization algorithm in detail. Recall that our regression problem is given as follows

$$\text{(P):} \quad \min_{\boldsymbol{\beta}\in\mathbb{R}^{(kr)}, \mathbf{V}\in\mathcal{V}} \|\mathbf{y} - \mathbf{S}(\mathbf{V})\boldsymbol{\beta}\|_2^2 + \lambda\boldsymbol{\beta}^T\mathbf{C}\boldsymbol{\beta} + \nu\boldsymbol{\beta}^T\boldsymbol{\beta}\,.$$

We focus on solving (P) for $\mathbf{V}$ with a fixed $\boldsymbol{\beta}$. We linearize the functions $s_t(\mathbf{v}_j^T\mathbf{x}_i)$ around $\mathbf{v}_j^0$

$$\begin{aligned} s_t(\mathbf{v}_j^T\mathbf{x}_i) &\approx s_t((\mathbf{v}_j^0)^T\mathbf{x}_i) + (\mathbf{v}_j - \mathbf{v}_j^0)^T\nabla_{\mathbf{v}}\, s_t(\mathbf{v}^T\mathbf{x}_i)\Big|_{\mathbf{v}=\mathbf{v}_j^0} \\ &= s_t((\mathbf{v}_j^0)^T\mathbf{x}_i) + (\mathbf{v}_j - \mathbf{v}_j^0)^T\mathbf{x}_i s_t'((\mathbf{v}_j^0)^T\mathbf{x}_i)\,. \end{aligned}$$

Plugging this approximation in $\mathbf{S}_j(\mathbf{v}_j)$, we get

$$\begin{aligned} \mathbf{S}_j(\mathbf{v}_j) &= \begin{bmatrix} s_1(\mathbf{v}_j^T\mathbf{x}_1) & \dots & s_k(\mathbf{v}_j^T\mathbf{x}_1) \\ s_1(\mathbf{v}_j^T\mathbf{x}_2) & \dots & s_k(\mathbf{v}_j^T\mathbf{x}_2) \\ \vdots & \vdots & \vdots \\ s_1(\mathbf{v}_j^T\mathbf{x}_n) & \dots & s_k(\mathbf{v}_j^T\mathbf{x}_n) \end{bmatrix} \\ &\approx \mathbf{S}_j(\mathbf{v}_j^0) \\ &+ \begin{bmatrix} (\mathbf{v}_j - \mathbf{v}_j^0)^T\mathbf{x}_1 s_1'((\mathbf{v}_j^0)^T\mathbf{x}_1) & \dots & (\mathbf{v}_j - \mathbf{v}_j^0)^T\mathbf{x}_1 s_k'((\mathbf{v}_j^0)^T\mathbf{x}_1) \\ \vdots & \vdots & \vdots \\ (\mathbf{v}_j - \mathbf{v}_j^0)^T\mathbf{x}_n s_1'((\mathbf{v}_j^0)^T\mathbf{x}_n) & \dots & (\mathbf{v}_j - \mathbf{v}_j^0)^T\mathbf{x}_n s_k'((\mathbf{v}_j^0)^T\mathbf{x}_n) \end{bmatrix} \\ &= \mathbf{S}_j(\mathbf{v}_j^0) + \mathbf{S}_j'(\mathbf{v}_j^0) \odot ((\mathbf{X}(\mathbf{v}_j - \mathbf{v}_j^0))\mathbf{1}_{1\times k}) \\ &\triangleq \mathbf{S}_j(\mathbf{v}_j^0) + \tilde{\mathbf{S}}_j(\mathbf{v}_j)\,, \end{aligned}$$

where $\odot$ denotes the point wise matrix operation, the data matrix $\mathbf{X} = \begin{bmatrix}\mathbf{x_1} & \mathbf{x_2} & \cdots & \mathbf{x_n}\end{bmatrix}^T \in \mathbb{R}^{n\times p}$, and

$$\mathbf{S}_j'(\mathbf{v}_j^0) = \begin{bmatrix} s_1'((\mathbf{v}_j^0)^T\mathbf{x}_1) & \dots & s_k'((\mathbf{v}_j^0)^T\mathbf{x}_1) \\ \vdots & \vdots & \vdots \\ s_1'((\mathbf{v}_j^0)^T\mathbf{x}_n) & \dots & s_k'((\mathbf{v}_j^0)^T\mathbf{x}_n) \end{bmatrix} \in \mathbb{R}^{n\times k}.$$

We therefore obtain

$$\mathbf{S}(\mathbf{V}) \approx \mathbf{S}(\mathbf{V}^0) + \tilde{\mathbf{S}}(\mathbf{V})\,,$$

where $\tilde{\mathbf{S}}(\mathbf{V})$ is obtained by concatenating the different $\tilde{\mathbf{S}}_j(\mathbf{v}_j)$. Then, we have $\mathbf{S}(\mathbf{V})\boldsymbol{\beta} \approx \mathbf{S}(\mathbf{V}^0)\boldsymbol{\beta} + \sum_{j=1}^r \tilde{\mathbf{S}}_j(\mathbf{v}_j)\boldsymbol{\beta}_j$, where $\boldsymbol{\beta}_j$ denotes the vector of length k whose entries represent the coefficients of the jth transfer function. Note that for any j, we have

$$\begin{aligned} \tilde{\mathbf{S}}_j(\mathbf{v}_j)\boldsymbol{\beta}_j &= \begin{bmatrix}\beta_{j1}\mathbf{I}_{n\times n} \cdots & \beta_{jk}\mathbf{I}_{n\times n}\end{bmatrix} \mathrm{vec}(\tilde{\mathbf{S}}_j(\mathbf{v}_j)) \\ &= \begin{bmatrix}\beta_{j1}\mathbf{I}_{n\times n} & \cdots & \beta_{jk}\mathbf{I}_{n\times n}\end{bmatrix} \cdot \\ &\qquad \left(\left(\mathrm{vec}(\mathbf{S}_j'(\mathbf{v}_j^0))\mathbf{1}_{1\times p}\right) \odot \left(\mathbf{1}_{k\times 1} \otimes \mathbf{X}\right)\right)(\mathbf{v}_j - \mathbf{v}_j^0) \\ &\triangleq \mathbf{M}_j(\mathbf{v}_j - \mathbf{v}_j^0), \end{aligned}$$

with $\otimes$ denoting the Kronecker product. Therefore, setting $\mathbf{M}$ to be equal to $[\mathbf{M}_1|\dots|\mathbf{M}_r]$, we get

$$\begin{aligned} \sum_{j=1}^r \tilde{\mathbf{S}}_j(\mathbf{v}_j)\boldsymbol{\beta}_j &= -\sum_{j=1}^r \mathbf{M}_j\mathbf{v}_j^0 + \sum_{j=1}^r \mathbf{M}_j\mathbf{v}_j \\ &= -\mathbf{M}\mathrm{vec}(\mathbf{V}^0) + \mathbf{M}\mathrm{vec}(\mathbf{V}) \\ &\triangleq \mathbf{b} + \mathbf{M}\mathrm{vec}(\mathbf{V}). \end{aligned}$$

Finally, we solve the following approximate problem, when $\boldsymbol{\beta}$ is fixed,

$$\min_{\mathbf{V}\in\mathcal{V}} \quad \|\tilde{\mathbf{y}} - \mathbf{M}\mathrm{vec}(\mathbf{V})\|_2^2,$$

where $\tilde{\mathbf{y}} = \mathbf{y} - \mathbf{S}(\mathbf{V}^0)\boldsymbol{\beta} - \mathbf{b}$.

References

Ba, A., Sinn, M., Goude, Y., and Pompey, P. (2012). Adaptive learning of smoothing functions: application to electricity load forecasting. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2519–2527.

Chen, Y. and Samworth, R. (2014). Generalised additive and index models with shape constraints. *arXiv preprint arXiv:1404.2957*.

Fan, S. and Hyndman, R. J. (2012). Short-term load forecasting based on a semi-parametric additive model. *IEEE Transactions on Power Systems*, 27(1):134–141.

Friedman, J. H. and Stuetzle, W. (1981). Projection pursuit regression. *Journal of the American Statistical Association*, 76:817–823.

Goude, Y., Nedellec, R., and Kong, N. (2014). Local short and middle term electricity load forecasting with semi-parametric additive models. *IEEE Transactions on Smart Grid*, 5(1):440–446.

Guyon, I. and Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning*, volume 2. Springer.

Hastie, T. J. and Tibshirani, R. J. (1990). *Generalized additive models*, volume 43. CRC Press.

Hu, J., Mitchell, J., Pang, J.-S., Bennett, K., and Kunapuli, G. (2008). On the global solution of linear programs with linear complementarity constraints. *SIAM Journal on Optimization*, 19(1):445–471.

Huang, J., Horowitz, J. L., and Wei, F. (2010). Variable selection in nonparametric additive models. *The Annals of Statistics*, 38(4):2282.

Jeroslow, R. (1978). Cutting planes for complementarity constraints. *SIAM Journal on Control and Optimization*, 16:56–62.

Jolliffe, I. (2005). *Principal component analysis*. Wiley Online Library.

Khatri, C. and Rao, C. R. (1968). Solutions to some functional equations and their applications to characterization of probability distributions. *Sankhyā: The Indian Journal of Statistics, Series A*, pages 167–180.

Lee, D. D. and Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791.

Morton, S. C. (1989). *Interpretable projection pursuit*. PhD Thesis, Stanford University, California.

Peng, R. D. and Welty, L. J. (2004). The NMMAPSdata package. *R News*, 4(2):10–14.

Ravikumar, P., Lafferty, J., Liu, H., and Wasserman, L. (2009). Sparse additive models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(5):1009–1030.

Sjöstrand, K., Clemmensen, L. H., Larsen, R., and Ersbøll, B. (2012). Spasm: A matlab toolbox for sparse statistical modeling. *Journal of Statistical Software Accepted for publication*.

Su, L. and Zhang, Y. (2013). Variable selection in nonparametric and semiparametric regression models. In *Handbook in Applied Nonparametric and Semi-Nonparametric Econometrics and Statistics*. Oxford University Press.

van der Maaten, L. J., Postma, E. O., and van den Herik, H. J. (2009). Dimensionality reduction: A comparative review. *Journal of Machine Learning Research*, 10(1-41):66–71.

Wang, X. and Brown, D. (2011). The spatio-temporal generalized additive model for criminal incidents. In *IEEE International Conference on Intelligence and Security Informatics (ISI)*, pages 42–47.

Wolsey, L. (1998). *Integer Programming*. Wiley Series in Discrete Mathematics and Optimization. Wiley.

Wood, S. (2006). *Generalized additive models: an introduction with R*. CRC press.

Yuan, M. (2011). On the identifiability of additive index models. *Statistica Sinica*, 21(4):1901.

Zhang, X., Liang, L., Tang, X., and Shum, H.-Y. (2008). L1 regularized projection pursuit for additive model learning. In *IEEE conference on computer vision and pattern recognition (CVPR)*, pages 1–8.

Zhao, T. and Liu, H. (2012). Sparse additive machine. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1435–1443.

Zhu, X., Huang, Z., Shen, H. T., Cheng, J., and Xu, C. (2012). Dimensionality reduction by mixed kernel canonical correlation analysis. *Pattern Recognition*, 45(8):3003–3016.

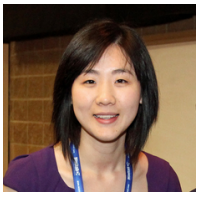
Zhu, X., Huang, Z., Yang, Y., Shen, H. T., Xu, C., and Luo, J. (2013). Self-taught dimensionality reduction on the high-dimensional small-sized data. *Pattern Recognition*, 46(1):215–229.

Zhu, X., Suk, H.-I., and Shen, D. (2014). Matrix-similarity based loss function and feature selection for alzheimer's disease diagnosis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3089–3096. IEEE.

Zou, H., Hastie, T., and Tibshirani, R. (2006). Sparse principal component analysis. *Journal of computational and graphical statistics*, 15(2):265–286.

Alhussein Fawzi received the M.Sc. degree in electrical and electronics engineering from the Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland in 2012. He is currently pursuing the PhD degree with the Signal Processing Laboratory (LTS4) at EPFL. His research interests include sparse signal and image processing, data mining and machine learning. He received twice the IBM PhD fellowship, in 2013 and 2015.

Jean-Baptiste Fiot is a Research Scientist at IBM Research - Ireland since December 2013. He received a Ph.D. degree in Applied Mathematics in 2013 from Paris Dauphine University in France, a Master degree in Applied Mathematics in 2009 from Ecole Nationale Superieure de Cachan in France, and a Master degree in Engineering in 2009 from Ecole Centrale Paris in France. Before joining IBM, he held Research positions in Paris Dauphine University in France, in Samsung Advanced Institute of Technology (SAIT) in South Korea, and in CSIRO - Australian e-Health Research Centre (AeHRC) in Australia. He was awarded the Best Student Paper Award in the VIPIMAGE 2011 conference, and the Thesis Prize 2014 of the Dauphine Foundation. His research interests include machine learning, signal and image processing, and optimization.



Bei Chen is a Research Staff Member in the Big Data Analytics & Systems department. She received her Ph.D. in Statistics from the University of Waterloo. Her current research interests include time series analysis, forecasting, resampling methods for dependent data and financial econometrics. Dr. Chen has more than 20 refereed publications in journals and international conferences.

Mathieu Sinn is a Research Staff Member and Manager in the Big Data Analytics & Systems department at the IBM Research laboratory in Dublin, Ireland. He received a Diploma in computer science in 2006, and a Ph.D. degree in mathematics in 2009, both from the University of Lbeck, Germany. Subsequently he was a Postdoctoral Research Fellow at the University of Waterloo, Canada, before joining IBM Research in 2011. His research interests lie at the intersection of statistics, machine learning and the analysis of real-world time series data. Dr. Sinn is the author or coauthor of 4 patents and more than 40 technical papers.

Pascal Frossard (S96,M01,SM04) received the M.S. and Ph.D. degrees, both in electrical engineering, from the Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, in 1997 and 2000, respectively. Between 2001 and 2003, he was a member of the research staff at the IBM T. J. Watson Research Center, Yorktown Heights, NY, where he worked on media coding and streaming technologies. Since 2003, he has been a faculty at EPFL, where he heads the Signal Processing Laboratory (LTS4). His research interests include graph signal processing, image representation and coding, visual information analysis, and distributed signal processing and communications.

Dr. Frossard has been the General Chair of IEEE ICME 2002 and Packet Video 2007. He has been the Technical Program Chair of IEEE ICIP 2014 and EUSIPCO 2008, and a member of the organizing or technical program committees of numerous conferences. He has been an Associate Editor of the IEEE TRANSACTIONS ON SIGNAL PROCESSING (2015-), IEEE TRANSACTIONS ON BIG DATA (2015-), IEEE TRANSACTIONS ON IMAGE PROCESSING (2010-2013), the IEEE TRANSACTIONS ON MULTIMEDIA (2004-2012), and the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY (2006-2011). He is the Chair of the IEEE Image, Video and Multidimensional Signal Processing Technical Committee (2014-2015), and an elected member of the IEEE Visual Signal Processing and Communications Technical Committee (2006-) and of the IEEE Multimedia Systems and Applications Technical Committee (2005-). He has served as Steering Committee Chair (2012-2014) and Vice-Chair (2004-2006) of the IEEE Multimedia Communications Technical Committee and as a member of the IEEE Multimedia Signal Processing Technical Committee (2004-2007). He received the Swiss NSF Professorship Award in 2003, the IBM Faculty Award in 2005, the IBM Exploratory Stream Analytics Innovation Award in 2008 and the IEEE Transactions on Multimedia Best Paper Award in 2011.