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 univariate transfer functions (also called ridge functions) $f_j: \mathbb{R} \rightarrow \mathbb{R}$ of the covariates: $y = \sum_{j=1}^{p} f_j(x_j) + \epsilon$. Here, $y$ is a real-valued response variable, $x = (x_1, \ldots, x_p)^T$ is a $p$-dimensional vector of covariates and $\epsilon$ 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 methods for additive models have been extensively studied in the literature (Su and Zhang, 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 Stuetzel, 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 Reduction for Additive Models (SDRAM), a framework for deriving covariates of additive models from high-dimensional 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 univariate transfer functions (also called ridge functions) $f_j: \mathbb{R} \rightarrow \mathbb{R}$ of the covariates: $y = \sum_{j=1}^{p} f_j(x_j) + \epsilon$. Here, $y$ is a real-valued response variable, $x = (x_1, \ldots, x_p)^T$ is a $p$-dimensional vector of covariates and $\epsilon$ 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 methods for additive models have been extensively studied in the literature (Su and Zhang, 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 Stuetzel, 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 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, \ldots, r\}. For any vector \( a = [a_1, \ldots, a_n]^T \in \mathbb{R}^n\), we denote by \(\text{supp}(a)\) the set \{\(i : a_i \neq 0\)\}, and use the notation \(a\{g\}\) to denote the vector \([a_{g_1}, \ldots, a_{g_m}]^T\), for any \( g = \{g_1, \ldots, g_m\} \subseteq [n]\). For given \( n \) and \( g \subseteq [n] \), we let \( \bar{g} \) denote the complement of \( g \). We use \( \|a\|_p \) to denote the \( \ell_p \) norm of \( a \). For any matrix \( A \in \mathbb{R}^{n_1 \times n_2} \), we denote by \( \text{vec}(A) \) the vector of size \( n_1n_2 \) obtained by stacking the columns of \( A \).

2.2 ADDITIVE INDEX MODELS

We consider the non-linear regression task

\[
y_i = g(x_i) + \epsilon_i,
\]

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

\[
g(x) = \mu + \sum_{j=1}^{r} f_j(v_j^T x),
\]

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 \( v_j \in \mathbb{R}^p \) are unknown weight vectors. Hence, the regression model has the form of an additive model applied to the derived covariates \( v_j^T x \) rather than to the “raw” input variables \( x \). 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 \( v_j \), 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 \(\{v_j\}_{j=1}^{r}\) 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 \( \mathcal{G} = \{g_1, \ldots, g_L\} \) be a set of \( L \) pairwise disjoint subsets of \{1, \ldots, \( p \)\}. Then,

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

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

\[
\forall j \in [r], \quad \|v_j\|_1 = 1, \quad v_j \geq 0.
\]

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

\[
\forall j, k \in [r], j \neq k, \quad \text{supp}(v_j) \cap \text{supp}(v_k) = \emptyset.
\]

The constraint \((C1)\) allows for partitioning the inputs into different user-specified groups, e.g., each containing variables of the same physical or logical type. The derived covariates are then constrained to combine solely input variables from the same group, hence facilitating a meaningful interpretation. Note that setting \( \mathcal{G} = \{g\} \), with \( g = \{1, \ldots, \( p \)\} \) corresponds to imposing no groups, as \((C1)\) is then satisfied for any weight vector \(\{v_j\}\). We assume that the desired number of derived variables \(r_i\) for each group is given and satisfies \( \sum_{i=1}^{L} r_i = r \). Moreover, \( \mathcal{G} \) is assumed to be known throughout this paper. \((C2)\) 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, which facilitates the interpretation compared to existing approaches which only impose a unit \( \ell_2 \) norm on the weight vectors. Finally, the disjoint support constraint in \((C3)\) prevents input variables from contributing to more than one derived covariate, thereby disentangling the different “causes” that generate the data, which is beneficial again both for the prevention of overfitting and facilitating the model interpretation.

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

\[
\mathcal{V} = \{\mathcal{V} = [v_1] \ldots [v_r] \mid \text{such that } \{v_j\}_{j=1}^{r} \text{ satisfy } (C1), (C2) \text{ and } (C3)\}.
\]

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

2.4 MODELIDENTIFIABILITY

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** (Identifiability). Assume that there exist \( \{(f_j, v_j)\}_{1 \leq j \leq r} \) and \( \{(h_j, w_j)\}_{1 \leq j \leq s} \) such that

\[
\forall x \in \mathbb{R}^p, \quad \mu + \sum_{j=1}^{r} f_j(v_j^T x) = \nu + \sum_{j=1}^{s} h_j(w_j^T x),
\]

where \( \{v_j\}_{1 \leq j \leq r} \) and \( \{w_j\}_{1 \leq j \leq s} \) 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 \( \pi : [r] \rightarrow [r] \) such that

\[
\forall j \in [r], \quad \begin{cases} f_j = h_{\pi(j)} \\ v_j = w_{\pi(j)} \end{cases}.
\]

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

\[
\min_{\forall v \in V} \sum_{i=1}^{n} \left( y_i - \sum_{j=1}^{r} f_j(v_j^T x_i) \right)^2 + \Omega(f_1, \ldots, f_r),
\]

where \( F \) is a predefined functional space and \( \Omega \) 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

\[
\forall j \in [r], \quad f_j(z) = \sum_{t=1}^{k} s_t(z) \beta_{jt},
\]

where \( s_t : \mathbb{R} \rightarrow \mathbb{R} \) denotes the \( t \)-th B-spline basis function, \( \beta_{jt} \) 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 \( \beta_j \) fully specify the transfer functions. Rewriting the problem in matrix form, we obtain the following constrained least-squares problem

\[
\min_{\beta \in \mathbb{R}^{kr}, \forall v \in V} \| y - S(V)\beta \|^2 + \Omega(\beta),
\]

with

\[
\beta = \begin{bmatrix} \beta_1^T \ldots \beta_r^T \end{bmatrix}^T, \quad \text{and} \quad S(V) = \left[ S_1(v_1) \ldots S_r(v_r) \right] \in \mathbb{R}^{n \times kr}
\]

We choose the regularization function

\[
\Omega(\beta) = \Omega_{\text{ridge}}(\beta) + \Omega_{\text{smooth}}(\beta)
\]

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

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

with the matrix \( C = (s''_t(x)s''_s(x)dx)_{i,j} \) and \( \lambda > 0 \). Note that the ridge regularization term favors vectors \( \beta \) 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

\[
(P): \min_{\beta \in \mathbb{R}^{kr}, \forall v \in V} \| y - S(V)\beta \|^2 + \lambda \beta^T C \beta + \nu \beta^T \beta.
\]

### 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 \( V \) is involved nonlinearly in the least-squares objective function. Moreover, the constraint \( V \in V \) imposes new difficulties compared to the unconstrained fitting problem. We propose an alternating iterative method, where we estimate sequentially the coefficient vector \( \beta \) and the weight matrix \( V \).

We begin by noting that, for a fixed \( V \), (P) reduces to a linear least squares problem that can be solved efficiently. The problem of finding \( V \) for a fixed coefficient vector \( \beta \), however, is much more challenging. Following a Gauss-Newton approach, we linearize the functions \( s_i(v_j^T x_i) \) around the current estimates \( v_j \) as

\[
s_i(v_j^T x_i) \approx s_i((v_j^0)^T x_i) + (v_j - v_j^0)^T \nabla_v s_i(v_j^T x_i) \bigg|_{v=v_j^0}.
\]

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

\[
\tilde{S}(V) = S(V^0) + \tilde{\dot{S}}(V),
\]

where \( \tilde{S}(V) \) is a matrix that can be written as a linear function of the weight vectors. Therefore, for any fixed vector \( \beta \), there exist a matrix \( M \) and a vector \( b \) not depending on \( V \) such that \( \tilde{S}(V)\beta = b + M \text{vec}(V) \). The detailed derivations
Algorithm 1 SDRAM learning algorithm

1. Initialize the entries of $V$ randomly using iid draws from a uniform distribution on $[0, 1]$, and divide by the sum of the weights to satisfy (C2).
2. For $m = 1, \ldots, N$,
   2.1 Update $\beta$ by solving
   $$(S(V)^T S(V) + \lambda C + \nu I) \beta = S(V)^T y.$$ 
   2.2 Update $V$ 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 BASELINE METHODS AND PERFORMANCE METRICS

We compare our model to the following baseline methods:

- **Additive Models (AM)**: with the following variants: AM 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 an 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.

- **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 (Sjöström 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
  $${\text{RMSE}} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2},$$
  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
  $$E(V, V^{GT}) = \frac{1}{p} \sum_{j=1}^{p} \sum_{l=1}^{r} |v_{jl} - v_{jl}^{GT}|,$$
  where $V$ and $V^{GT}$ are respectively the estimated and ground truth weight matrices.

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 $V$.

4.2 TOY EXAMPLE

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

$$y_i = f_1(0.5x_{i,1} + 0.25x_{i,2} + 0.25x_{i,3}) + f_2(x_{i,4}) + f_3(0.5x_{i,5} + 0.5x_{i,6}) + \epsilon_i,$$  \hspace{1cm} (7)

where $f_1(x) = 2\exp(x)$, $f_2(x) = 2\sin(\pi x)$, $f_3(x) = 10x^2$, the error terms $\epsilon_i$ are iid samples from a standard normal distribution, and the covariates $x_{1,\ldots,6}$ 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 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. Finally, all experiments of this paper are performed on a laptop with an Intel i7 CPU, on which the mixed-integer program took less than 1 min to solve.

4.3 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, 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

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

and use our algorithm to derive two covariates from the “Lagged availability” group (i.e., we set $r_1 = r_2 = r_3 = 1$, $r_4 = 3$),


4. [http://www.dublinbikes.ie](http://www.dublinbikes.ie)
and $r_1 = 2$). 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. Finally, SDRAM significantly outperforms unsupervised dimensionality reduction approaches PCA+AM, and SPCA+AM. The paired Wilcoxon test shows that the improvement of SDRAM over all methods is statistically significant at a significance level of 0.01.

Figure 3 displays the weight matrices obtained using SDRAM, PPR and PCA+AM for one particular bike station (Station 1). While PPR and PCA 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 the competing methods are dense. Note that PPR and PCA also 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 associated transfer functions. In the above 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 which allows to disentangle positive and negatively correlated stations. For this example, there is moreover an intuitive explanation of the derived features: the bike station for which the predictions are computed lies in the commercial 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

![Fig. 3. Weight matrix $V$ learned using SDRAM (left), PPR (middle) and PCA (right). 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.](image)

![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.](image)
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 transportation hub to complete the last mile of their journey to the university campus.

4.4 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 explicitly 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 and temperature data from 40 weather stations from MADIS. 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 \(T_{mean}\) (the mean over the past seven days for each station). We enforce the following groups in the derivation of the covariates:

\[ G = \{ \{ \text{"Time of Year"}, \{ \text{"Time of Day"}, \{ \text{"Day of Week"}, \{ \text{"Lag load"}, \{ T \}, \{ T_{lag24}\}, \{ T_{mean24}\}, \{ T_{min24}\}, \{ T_{max24}\}, \{ T_{mean}\} \} \} \} \} \]

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. AM1 and PPR 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. 5). We have observed that PPR strongly overfits the data when \( r \geq 3 \), leading to very poor testing accuracy. Hence, the constraints on \( V \) 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. 6 shows the maps of the weights associated with the different weather stations in the derivation of the temperature-related covariates. Interestingly, for most
TABLE 2
Model accuracy (RMSE) and complexity on the electric load forecasting problem.

<table>
<thead>
<tr>
<th>Method</th>
<th>RMSE</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Training</td>
<td>Testing</td>
</tr>
<tr>
<td>SDRAM</td>
<td>26.6</td>
<td>27.5</td>
</tr>
<tr>
<td>PPR</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Additive models</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SpAM</td>
<td>26.0</td>
<td>28.1</td>
</tr>
<tr>
<td>PCA + AM</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPCA + AM</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fig. 5. RMSE of the PPR method as a function of the number of derived covariates.

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.\(^8\)

5 CONCLUSION

We proposed a novel framework for learning additive models with a moderate number of covariates derived from a potentially very 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.

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.

\[ \forall x \in \mathbb{R}^p, \quad f(v^T x) = h(w^T x), \]

with \(v\) and \(w\) non-negative vectors with unit \(\ell_1\) norm. Then \(v = w\) and \(f = h\).

Proof. We proceed by contradiction, and assume that \(v\) and \(w\) are not collinear. In other words, assume that \(\text{span}(v) \neq \text{span}(w)\), which is equivalent to \(\text{span}(v)^\perp \neq \text{span}(w)^\perp\). Moreover, \(\text{span}(v)^\perp\) is not strictly included in \(\text{span}(w)^\perp\), as both subspaces have the same dimension. Therefore, there exists \(x_0\) such that \(x_0 \perp v\) and \(x_0 \not\in \text{span}(w)^\perp\). In other words, \(v^T x_0 = 0\), and \(w^T x_0 \neq 0\). For any \(\mu \in \mathbb{R}\), we therefore have

\[ f(v^T (\mu x_0)) = f(\mu v^T x_0) = f(0) = 0 \quad \Rightarrow \quad h(w^T (\mu x_0)) = h(\mu w^T x_0) \quad \neq 0. \]

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

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

\(^8\) http://www.nws.noaa.gov/climate/local_data.php?wfo=BTV, see Vermont Annual Mean High/Low Temperature
Proposition A.2. Let \( \{v_j\}_{j=1}^r \) and \( \{w_j\}_{j=1}^s \) be weight vectors that satisfy constraints (C1), (C2) and (C3). Suppose that

\[
\forall x \in \mathbb{R}^p, \quad \sum_{j=1}^r f_j(v_j^T x) = \sum_{j=1}^s h_j(w_j^T x),
\]

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 \( \pi \) such that, for all \( j \in [r] \)

\[
v_j = w_{\pi(j)}, \quad f_j = h_{\pi(j)}. \tag{11}
\]

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

\[
\forall j \in [r], \quad \exists \pi(j) \text{ such that } \text{supp}(v_j) = \text{supp}(w_{\pi(j)}). \tag{12}
\]

Indeed, if Eq. (12) holds, then by evaluating Eq. (10) at \( x \) such that \( \text{supp}(x) = \text{supp}(v_j) \), we get

\[
f_j(v_j^T x) = h_{\pi(j)}(w_{\pi(j)}^T x),
\]

where we used the disjoint supports assumption and the fact that \( f_j(0) = h_j(0) = 0 \) for all \( j, k \). The above equality generalizes to any \( x \in \mathbb{R}^p \) as \( v_j \) and \( w_{\pi(j)} \) have zero entries outside of \( \text{supp}(v_j) \)

\[
\forall x \in \mathbb{R}^p, \quad f_j(v_j^T x) = h_{\pi(j)}(w_{\pi(j)}^T x). \tag{13}
\]

We therefore obtain from Proposition A.1 that \( v_j = w_{\pi(j)} \) and \( f_j = h_{\pi(j)} \). Note moreover that \( \pi \) is one-to-one as \( \pi(j_1) = \pi(j_2) \) would imply \( \text{supp}(v_{j_1}) = \text{supp}(v_{j_2}) \) 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 \( \bigcup_j \text{supp}(v_j) \times \text{supp}(v_j) = \bigcup_j \text{supp}(w_j) \times \text{supp}(w_j) \), from which we can see that the supports of \( v_j \) and \( w_j \) 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

\[
\forall j \in [s], \quad \text{supp}(v_{j_0}) \neq \text{supp}(w_j). \tag{14}
\]

Then,

\[
\forall x \in \mathbb{R}^p, \quad \text{supp}(x) = \text{supp}(v_{j_0}) \quad \text{such that } \text{supp}(v_{j_0}) = \text{supp}(w_{j_1}), \quad f_{j_0}(v_{j_0}^T x) = \sum_{j=1}^s h_j(w_j^T x). \tag{15}
\]

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 \( a v_{j_0}^T x = b w_j^T x \) for all \( x \) with the same support as \( v_{j_0} \) for some \( a, b \), and index \( j_1 \). If \( \text{supp}(v_{j_0}) \notin \text{supp}(w_{j_1}) \), then there exists \( k \) such that \( v_{j_0 k} \neq 0 \) and \( w_{j_1 k} = 0 \). By setting \( x = e_k \), 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 \( \text{supp}(v_{j_0}) \subset \text{supp}(w_{j_1}) \). In that case, we have
\[ bw_j^T x = \alpha v_j^T x \] for all \( x \) such that \( \text{supp}(x) = \text{supp}(w_j) \), 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 \( \text{supp}(v_{j_0}) \not\subseteq \text{supp}(w_j) \) for all \( j \). If Eq. \((15)\) is to hold, there exists at least one \( j \) such that \( \text{supp}(v_{j_0}) \cap \text{supp}(w_j) \neq \emptyset \), and let \( j_1 \) be such an index. Denote by \( k \) an element in \( \text{supp}(v_{j_0}) \cap \text{supp}(w_{j_1}) \). Since \( \text{supp}(v_{j_0}) \not\subseteq \text{supp}(w_{j_1}) \), there exists an element \( l \in \text{supp}(v_{j_0}) \) and not in \( \text{supp}(w_{j_1}) \). Therefore, the cross-term \( x_k \) belongs to the left hand side of Eq. \((15)\), but not to the right hand side. This cannot hold, and we conclude that \( \text{supp}(v_{j_0}) \subseteq \text{supp}(w_{j_1}) \). Since \( \text{supp}(v_{j_0}) \) and \( \text{supp}(w_{j_1}) \) have the same cardinality, we must have \( \text{supp}(v_{j_0}) = \text{supp}(w_{j_1}) \).

Now we do the change of variable \( z = W^T x \) and obtain
\[
\forall z \in \mathbb{R}^p, \quad f_1(u_j^T z) = \sum_{j=1}^s h_j(z_j). \tag{18}
\]

Setting \( z = z_k e_k \), we get
\[
\text{For } 1 \leq k \leq s: \quad f_1(u_k z_k) = h_k(z_k), \tag{19}
\]
\[
\text{For } s < k \leq p: \quad f_1(u_k z_k) = 0. \tag{20}
\]

Note that if \( u_{1s} = 0 \) 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 \( u_{1s} \neq 0 \) 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 \( u_1 = (u_{11}, \ldots, u_{1s}, 0, \ldots, 0) \) with \( u_{11}, \ldots, u_{1s} \neq 0 \). Assuming \( s > 1 \), we take \( z = (z_1, z_2, 0, \ldots, 0) \) and using \((18)\) and \((19)\) we get
\[
\forall z_1, z_2 \in \mathbb{R}, \quad f_1(u_{11} z_1 + u_{12} z_2) = h_1(z_1) + h_2(z_2) = f_1(u_{11} z_1) + f_1(u_{12} z_2). \tag{21}
\]

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 \( u_1 = \lambda e_1 \) and \((18)\) becomes
\[
\forall z_1 \in \mathbb{R}, \quad f_1(\lambda z_1) = h_1(z_1). \tag{22}
\]

We also get \( v_1 = W_1 = \lambda W_1 = \lambda w_1 \). Since \( \|v_1\|_1 = \|w_1\|_1 = 1 \), we get \( \lambda = \pm 1 \). The positivity of \( v_1 \) and \( w_1 \) gives \( \lambda \geq 1 \), and thus \( v_1 = w_1 \) and \( f_1 = h_1 \).

A.2.2 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 \( \nu_{r+1} = w_s \) (up to a permutation of the terms), and \((H_r) \) holds. Calling \((P_r) \) would then terminate the proof of \((F_{r+1}) \).

The same change of variable as in the initialization gives
\[
\forall z \in \mathbb{R}^p, \quad \sum_{j=1}^{r+1} f_j(u_j^T z) = \sum_{j=1}^s h_j(z_j). \tag{23}
\]

First case: all \( \{u_j\}_{1 \leq j \leq r+1} \) 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 \( u_1 \) with one non-zero entry. Without loss of generality, we can say this \( u_1 \) is \( u_{1s} \), and with a permutation of coordinates take \( u_{1s+1} = \lambda e_s \) for some \( \lambda \). Using the equality \( v_{r+1} = W_{u_{r+1}} = \lambda w_s \) and the unit norm constraints on \( v_{r+1} \), we get \( \lambda = 1 \). We therefore have \( u_{r+1} = e_s \) and \( v_{r+1} = w_s \). Note also that
\[
\forall j \in [r], \quad v_j^T v_{r+1} = v_j^T w_s = (W_{u_1})^T w_s = u_j^T e_s \|w_s\|^2 = u_j s \|w_s\|^2, \tag{24}
\]

where we have used the fact that the columns of \( W \) are orthogonal. Since the \( v_j \)s are orthogonal to each other, we have \( v_j^T v_{r+1} = 0 \), and therefore \( u_{j} s = 0 \) for all \( j \in [r] \) as \( w_s \)
is a nonzero vector. By rewriting Eq. (23) and setting $z_x = 0$, we have
\[ \forall z \in \mathbb{R}^p, \quad z_x = 0 \}
\]
\[ \sum_{j=1}^r f_j(u^T_x z) = \sum_{j=1}^{s-1} h_j(z_j). \quad (25) \]
Moreover, since $h_{x_j} = 0$ for all $j \in [r]$, the above equality is valid for all $z \in \mathbb{R}^p$, and we have
\[ \forall z \in \mathbb{R}^p, \quad \sum_{j=1}^r f_j(u^T_x z) = \sum_{j=1}^{s-1} h_j(z_j). \quad (26) \]
Using the change of variables, we therefore get
\[ \forall x \in \mathbb{R}^p, \quad \sum_{j=1}^r f_j(v^T_j x) = \sum_{j=1}^{s-1} h_j(w^T_j x). \quad (27) \]
which corresponds to $(H_r)$. By calling $(P_r)$, we have $r = s-1$ and there exists a permutation $\pi : [r] \rightarrow [r]$ such that
\[ \forall j \in [r], \quad f_j = h_{x_{\pi(j)}}, \quad v_j = w_{\pi(j)}. \quad (28) \]
We therefore have $f_{r+1}(v^T_{r+1} x) = h_s(w^T_s x)$, and using the equality $v_{r+1} = w_s$, 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
\[ \forall x \in \mathbb{R}^3, \quad \frac{1}{2} \left( \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 1 \end{bmatrix} x \right) = \left( \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 1 \end{bmatrix} x \right) + \frac{1}{2} \left( \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} x \right). \quad (29) \]
The above model is unidentifiable as the left-hand side and right-hand side models are equal for all $x \in \mathbb{R}^3$, 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
\[(P): \min_{\beta \in \mathbb{R}^{k_x}, \nu \in \mathcal{V}} \|y - S(V) \beta\|_2^2 + \lambda \beta^T C \beta + \nu \beta^T \beta. \]
We focus on solving $(P)$ for $V$ with a fixed $\beta$. We linearize the functions $s_t(v^T_j x_i)$ around $v^0_j$
\[ s_t(v^T_j x_i) \approx s_t((v^0_j)^T x_i) + (v_j - v^0_j)^T \nabla v_s(v^T_j x_i) \big|_{v=v^0_j} \]
\[ = s_t((v^0_j)^T x_i) + (v_j - v^0_j)^T x_i, s'_t((v^0_j)^T x_i). \]

Plugging this approximation in $S_j(v_j)$, we get
\[
S_j(v_j) = \begin{bmatrix} s_1(v^0_j x_1) & \ldots & s_k(v^0_j x_1) \\ s_1(v^0_j x_2) & \ldots & s_k(v^0_j x_2) \\ \vdots & \ddots & \vdots \\ s_1(v^0_j x_n) & \ldots & s_k(v^0_j x_n) \end{bmatrix} \\
\quad \approx S_j(v^0_j) + \left( (v_j - v^0_j)^T x_1 s'_1((v^0_j)^T x_1) \right) \ldots \left( (v_j - v^0_j)^T x_n s'_k((v^0_j)^T x_n) \right) \\
\quad \quad + \left( (v_j - v^0_j)^T x_1 v'_1((v^0_j)^T x_1) \right) \ldots \left( (v_j - v^0_j)^T x_n v'_k((v^0_j)^T x_n) \right) \\
\quad \quad = S_j(v^0_j) + S'_j(v^0_j) \bigcirc(K(x_j - v^0_j)I_{1 \times k}) \\
\quad \quad \triangleq S_j(v^0_j) + \hat{S}_j(v_j),
\]
where $\bigcirc$ denotes the point-wise matrix operation, the data matrix $X = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^T \in \mathbb{R}^{n \times p}$, and
\[ S'_j(v^0_j) = \begin{bmatrix} s'_1((v^0_j)^T x_1) & \ldots & s'_k((v^0_j)^T x_1) \\ \vdots & \ddots & \vdots \\ s'_1((v^0_j)^T x_n) & \ldots & s'_k((v^0_j)^T x_n) \end{bmatrix} \in \mathbb{R}^{n \times k}. \]
We therefore obtain
\[ S(V) \approx S(V^0) + \hat{S}(V), \]
where $\hat{S}(V)$ is obtained by concatenating the different $\hat{S}_j(v_j)$. Then, we have $S(V) \beta \approx S(V^0) \beta + \sum_{j=1}^{r} S_j(v_j) \beta_j$, where $\beta_j$ denotes the vector of length $k$ whose entries represent the coefficients of the $j$th transfer function. Note that for any $j$, we have
\[ \hat{S}_j(v_j) \beta_j = \begin{bmatrix} \beta_{j1} I_{n \times n} & \cdots & \beta_{jk} I_{n \times n} \end{bmatrix} \text{vec}(S_j(v_j)) = \begin{bmatrix} \beta_{j1} I_{n \times n} & \cdots & \beta_{jk} I_{n \times n} \end{bmatrix} \cdot \left( (\text{vec}(S'_j(v^0_j))) (I_{1 \times p}) \bigcirc (I_{k \times 1} \otimes X) \right) (v_j - v^0_j) \]
\[ \triangleq M_j(v_j - v^0_j), \]
with $\bigcirc$ denoting the Kronecker product. Therefore, setting $M$ to be equal to $[M_1] \ldots [M_r]$, we get
\[ \sum_{j=1}^{r} \hat{S}_j(v_j) \beta_j = - \sum_{j=1}^{r} M_j v^0_j + \sum_{j=1}^{r} M_j v_j = - M \text{vec}(V^0) + M \text{vec}(V) \]
\[ \triangleq b + M \text{vec}(V). \]
Finally, we solve the following approximate problem, when $\beta$ is fixed,
\[ \min_{\nu \in \mathcal{V}} \|\tilde{y} - M \text{vec}(V)\|_2^2, \]
where $\tilde{y} = y - S(V^0) \beta - b$.

### REFERENCES


**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 Superiéure 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 Post-doctoral 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.