






## DDAC-SpAM: A Distributed Algorithm for Fitting High-dimensional Sparse Additive Models with Feature Division and Decorrelation

Yifan He, Ruiyang Wu, Yong Zhou & Yang Feng


**To cite this article:** Yifan He, Ruiyang Wu, Yong Zhou & Yang Feng (2024) DDAC-SpAM: A Distributed Algorithm for Fitting High-dimensional Sparse Additive Models with Feature Division and Decorrelation, Journal of the American Statistical Association, 119:547, 1933-1944, DOI: [10.1080/01621459.2023.2225743](https://doi.org/10.1080/01621459.2023.2225743)


**To link to this article:** <https://doi.org/10.1080/01621459.2023.2225743>

 [View supplementary material](#) 


 Published online: 26 Jul 2023.

 [Submit your article to this journal](#) 

 Article views: 792

 [View related articles](#) 

 [View Crossmark data](#) 

 Citing articles: 1 [View citing articles](#) 



# DDAC-SpAM: A Distributed Algorithm for Fitting High-dimensional Sparse Additive Models with Feature Division and Decorrelation

Yifan He<sup>a</sup>, Ruiyang Wu<sup>ib</sup>, Yong Zhou<sup>ib</sup>, and Yang Feng<sup>ib</sup>

<sup>a</sup>Department of Statistics, The Chinese University of Hong Kong, Hong Kong; <sup>b</sup>Department of Biostatistics, School of Global Public Health, New York University, New York, NY; <sup>c</sup>Academy of Statistics and Interdisciplinary Sciences and School of Statistics, East China Normal University, Shanghai, China

## ABSTRACT

Distributed statistical learning has become a popular technique for large-scale data analysis. Most existing work in this area focuses on dividing the observations, but we propose a new algorithm, DDAC-SpAM, which divides the features under a high-dimensional sparse additive model. Our approach involves three steps: divide, decorrelate, and conquer. The decorrelation operation enables each local estimator to recover the sparsity pattern for each additive component without imposing strict constraints on the correlation structure among variables. The effectiveness and efficiency of the proposed algorithm are demonstrated through theoretical analysis and empirical results on both synthetic and real data. The theoretical results include both the consistent sparsity pattern recovery as well as statistical inference for each additive functional component. Our approach provides a practical solution for fitting sparse additive models, with promising applications in a wide range of domains. Supplementary materials for this article are available online.

## ARTICLE HISTORY

Received October 2021  
Accepted June 2023

## KEYWORDS

Additive model; Consistency; Decorrelate and conquer; Divide; Feature space partition; Variable selection

## 1. Introduction

In modern statistics and computing practices, there exists a common bottleneck that complex data with unprecedented size cannot fit into memory nor be analyzed within reasonable time on a single machine. One popular solution is distributed statistical learning which works by distributing the learning task to different machines and combining the estimates afterward (Boyd et al. 2011; Zhang, Wainwright, and Duchi 2012; Lee et al. 2017). According to the way of partitioning the dataset, we call a method *observation-distributed* or *feature-distributed*. Substantial progress has been made on the former type that partitions observations into subsets and then fits the same model using each subset with the same features in different machines. Most of the literature focuses on the massive linear or generalized linear models (Chen and Xie 2014; Zhang, Duchi, and Wainwright 2015; Zeng and Lin 2015; Tang, Zhou, and Song 2016; Zhao, Cheng, and Liu 2016; He et al. 2016; Lee et al. 2017; Battey et al. 2018; Shi, Lu, and Song 2018). For nonparametric inference, the existing studies include a partial linear model (Zhao, Cheng, and Liu 2016) and nonparametric regression model (Zhang, Duchi, and Wainwright 2013). However, to the best of our knowledge, the feature-distributed statistical learning method, especially for high dimensional nonparametric regression, still remains to be developed.

In this article, we propose a *feature-distributed* algorithm for sparse additive model with potentially massive number of covariates ( $p \gg n$ ). This problem can be formulated as follows:

we are given  $n$  observations with response  $y_i \in \mathbb{R}$  and covariates  $\{x_{i1}, \dots, x_{ip}\} \in \mathbb{R}^p$  for  $i = 1, \dots, n$ . The goal is to fit the additive model (Stone 1985)

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

where the number of covariates  $p$  can grow much faster than the sample size  $n$  with  $\log(p) = n^\nu$  for some  $\nu \in (0, 1)$ . What makes the high dimensional inference possible is the sparsity assumption where only a small subset of  $\{f_j, j = 1, \dots, p\}$  are nonzero functions. Many sparsity-promoted estimators have been proposed for (1) (Aerts, Claeskens, and Wand 2002; Ravikumar et al. 2009; Meier, Van de Geer, and Bühlmann 2009; Koltchinskii and Yuan 2010; Huang, Horowitz, and Wei 2010; Raskutti, Wainwright, and Yu 2012; Yuan and Zhou 2016; Petersen, Witten, and Simon 2016; Sadhanala and Tibshirani 2019). Since sparse additive models (SpAM) are essentially a functional version of the group lasso, Ravikumar et al. (2009) borrowed ideas from the sparse linear model and proposed a corresponding algorithm for solving the problem

$$\min_{\beta_j \in \mathbb{R}^{d_n}, j=1, \dots, p} \frac{1}{2n} \left\| Y - \sum_{j=1}^p \Psi_j \beta_j \right\|_2^2 + \lambda \sum_{j=1}^p \sqrt{\frac{1}{n}} \beta_j^T \Psi_j^T \Psi_j \beta_j,$$

where  $\beta_j = (\beta_{j1}, \dots, \beta_{jd_n})^T$  is a length- $d_n$  vector of coefficients, and  $\Psi_j = (\psi_{j1}, \dots, \psi_{jd_n})$  is an  $n \times d_n$  matrix of the truncated

set of orthogonal basis functions for  $f_j$  evaluated at the training data. Minimax optimal rates of convergence were established in Raskutti, Wainwright, and Yu (2012) and Yuan and Zhou (2016). Other extensions of the high-dimensional additive model have been proposed by Lou et al. (2016), Sadhanala and Tibshirani (2019), Petersen and Witten (2019), and Haris, Simon, and Shojaie (2022).

Considering datasets of massive dimensions that the computational complexity or memory requirements cannot fit into one single computer, the aforementioned frameworks for additive models are not directly applicable. A distributed solution and the decentralized storage of datasets are necessary. Most works on distributed statistical inference assume that the data is partitioned by observations, because of the good theoretical properties of the averaged estimators. However, under the high-dimensional additive model, the spline basis functions are constructed from the whole sample. Besides, each component is represented by a group of basis functions which results in an even higher dimension of the design matrix than the number of observations. It is thus desirable to seek feature-distributed algorithms. But feature-distributed studies are scarce, partially due to the fact that the feature distribution process which ignores the correlation between covariates could lead to incorrect inference. Indeed, dividing feature space directly usually leads to misspecified models and ineradicable bias.

Next, we review several works on feature distributed methods. Inspired by *group testing*, Zhou et al. (2014) proposed a *parallelizable feature selection* algorithm. They randomly sectionalized features repeatedly for tests and then ranked the features by the test scores. This attempt can boost efficiency but its success heavily depends on the correlation structure of covariates. Song and Liang (2015) proposed a Bayesian variable selection approach for ultrahigh dimensional linear regression models based on splitting feature set into lower-dimensional subsets and screening important variables, respectively, with the marginal inclusion probability for final aggregation. Similar treatments can be found in the Yang et al. (2016), although in the final stage they used the *sketch* approach for further selection. The efficiency of this kind of algorithms will again be highly affected by the correlation structure among features. Thus, the identifiability condition for controlling the degree of multicollinearity is necessary. Based on those key facts, Wang, Dunson, and Leng (2016) relaxed the correlation requirements by preprocessing the data with a *decorrelation* operator (DECO) to lower the correlation in feature space under the linear regression model. In a related work, this decorrelation operator was shown to satisfy the irrepresentable condition for *lasso* (Jia and Rohe 2015). With DECO, we can get consistent estimates of coefficients with misspecified submodels.

In this work, we consider decorrelating covariates and propose the feature-distributed algorithm DDAC-SpAM under the high-dimensional additive model. That is, we first divide the whole dataset by predictors, that is, each local machine operates on only  $p_i$  variables. Then local machines approximate each component in additive models with a truncated set of B-spline basis. After decorrelating the design matrix of the B-spline basis with the central machine, local machines can in parallel conduct group lasso fit efficiently. Finally, the central machine combines

the discovered important predictors and refines the estimates. This algorithm can be regarded as a functional extension of the DECO procedure proposed by Wang, Dunson, and Leng (2016). It provides an efficient way of conducting simultaneous feature selection and point estimation. On top of that, we incorporate a debiasing step (Van de Geer et al. 2014; Cai, Zhang, and Zhou 2022) and propose a Chi-squared test for each functional summands in model (1).

The rest of this article is organized as follows. In Section 2, we review the sparse additive model problem. In Section 3, we introduce the distributed feature selection procedure for the additive model after decorrelation. In Section 4, we construct a Chi-squared test based on a debiased version of the DDAC-SpAM algorithm. In Section 5, we present the *sparsistency* property (i.e., sparsity pattern consistency) (Ravikumar et al. 2009) of the DDAC-SpAM algorithm and asymptotic theories for the hypothesis testing framework. Our simulations and a real data analysis are presented in Sections 6 and 7, respectively, showing the efficiency and effectiveness of our method. We conclude with a discussion in Section 8. All the technical details are relegated to the supplementary material.

Some standard notation used throughout this article is collected here. For number  $a$ ,  $\lceil a \rceil$  represent the smallest integer larger than or equal to  $a$ . For a square matrix  $A$ , let  $\lambda_{\min}(A)$ ,  $\lambda_{\max}(A)$  and  $\text{tr}(A)$  denote the minimum and maximum eigenvalues and the trace. We use the norms  $\|A\| = \sqrt{\lambda_{\max}(A^T A)}$ ,  $\|A\|_F = \sqrt{\text{tr}(A^T A)}$  and  $\|A\|_{\infty} = \max_i \sum_{j=1}^n |A_{ij}|$ . For vector  $v = (v_1, \dots, v_k)^T$ , we use the norms  $\|v\| = \sqrt{\sum_{j=1}^k v_j^2}$  and  $\|v\|_{\infty} = \max_j |v_j|$ . For function  $f$ ,  $f = 0$  means  $f$  is the zero constant function.

## 2. Sparse Additive Model

Given a random sample  $\{(x_{i1}, \dots, x_{ip}), y_i\}_{i=1}^n$ , where for each  $j$ ,  $\{x_{ij}, i = 1, \dots, n\} \stackrel{\text{iid}}{\sim} \mu_j$  in which  $\mu_j$  is a probability distribution supported on  $[0, 1]$ , we consider the nonparametric additive model

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

where the error  $\varepsilon_i \stackrel{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2)$ ,  $i = 1, \dots, n$ . Let  $\varepsilon = (\varepsilon_1, \dots, \varepsilon_n)^T$ .  $X = (x_{ij})_{n \times p} = (X_1, \dots, X_p)$  is the  $n \times p$  design matrix and  $Y = (y_1, \dots, y_n)^T$  is the response vector. To ensure identifiability of  $\{f_j, j = 1, \dots, p\}$ , we assume  $E f_j(x_{ij}) = 0$ .

For function  $f_j$ , let  $\{\psi_{jk}, k = 1, 2, \dots\}$  denote the uniformly bounded basis functions with respect to the Lebesgue measure on  $[0, 1]$ . Following Ravikumar et al. (2009), we assume the following smoothness condition.

**Condition 1.** For  $j = 1, \dots, p$ ,  $f_j \in \mathcal{S}_j$  where

$$\mathcal{S}_j = \left\{ f_j \in \mathcal{H}_j : f_j(x) = \sum_{k=1}^{\infty} \beta_{jk} \psi_{jk}(x), \sum_{k=1}^{\infty} \beta_{jk}^2 k^4 \leq C^2 \right\}$$

for some  $0 < C < \infty$ , where  $\mathcal{H}_j$  is a Hilbert space of mean zero square integrable functions with the inner product  $\langle f_j, f'_j \rangle =$

$Ef_j(x_{ij})f'_j(x_{ij})$ , that is,  $Ef_j(x_{ij}) = 0$ ,  $\|f_j\|^2 = \langle f_j, f_j \rangle < \infty$ , and  $\sup_x |\psi_{jk}(x)| \leq B$  for some  $B$ .  $\{\beta_{jk}, k = 1, 2, \dots\}$  are the parameters corresponding to  $f_j$ .

The standard form of the penalized additive model optimization problem is

$$\min_{f_1 \in \mathcal{S}_1, \dots, f_p \in \mathcal{S}_p} \sum_{i=1}^n \left\{ y_i - \sum_{j=1}^p f_j(x_{ij}) \right\}^2 + J(f_1, \dots, f_p). \quad (2)$$

where  $J$  is a sparsity-smoothness penalty. In this article, we restrict our discussion to the sparsity-inducing penalty

$$J(f_1, \dots, f_p) = \lambda_n \sum_{j=1}^p \sqrt{\sum_{i=1}^n f_j^2(x_{ij})}.$$

Following Meier, Van de Geer, and Bühlmann (2009), we approximate  $\{f_j, j = 1, \dots, p\}$  by a cubic B-spline with a proper number of knots. One possible choice would be to place  $d_n - 4$  interior knots at the empirical quantile of  $X_j$ , that is,

$$f_j(x) \approx f_{nj}(x) = \sum_{k=1}^{d_n} \beta_{jk} \psi_{jk}(x).$$

With Condition 1, we can bound the truncation bias by  $\|f_j - f_{nj}\|^2 = O(1/d_n^3)$ . Let  $h = \sum_{j=1}^p f_j$  and  $h_n = \sum_{j=1}^p f_{nj}$ . Let  $S = \{j : f_j \neq 0\}$  be the active set of variables and  $s = |S|$  be its cardinality. It follows that  $\|h - h_n\|^2 = O(s^2/d_n^3)$ .

Let  $\Psi_j$  denote the  $n \times d_n$  B-spline basis matrix for  $f_j$ , where  $\Psi_j(i, k) = \psi_{jk}(x_{ij})$ . Let  $\beta_j$  denote the corresponding coefficient vector  $(\beta_{j1}, \dots, \beta_{jd_n})$ . Then the optimization problem (2) can be reformulated as

$$\min_{\beta_1, \dots, \beta_p} \left\| Y - \sum_{j=1}^p \Psi_j \beta_j \right\|^2 + \lambda \sum_{j=1}^p \frac{1}{\sqrt{n}} \|\Psi_j \beta_j\|. \quad (3)$$

This group-wise variable selection problem can be solved by the standardized group lasso technique (Simon and Tibshirani 2012). The algorithm for standardized group lasso can be viewed as a special group lasso procedure after orthogonalization within each group, in which group lasso is computationally more intensive than lasso (Tibshirani 1996). Since its solution paths are not piecewise linear, the least angle regression (LARS) algorithm (Efron et al. 2004) is not applicable. Instead, the block coordinate-wise descent-type algorithms (Hastie and Tibshirani 1990; Meier, Van De Geer, and Bühlmann 2008; Foygel and Drton 2010; Wood 2011; Yang and Zou 2015) are common approaches. Computational complexity is somewhat tricky to quantify since it largely depends on the number of iterations. Since each spline block costs  $O(nd_n)$  operations,  $O(npd_n)$  calculations are required for entire data in one pass. The number of back-fitting loops required for convergence is usually related to  $p$ . As for the noniterative components, orthogonalization within each block can be solved by QR decomposition which costs  $O(npd_n^2)$  operations. Besides, compared with linear regression, memory footprint increases with the expanded spline basis functions  $\Psi_j$  taking place of original  $X_j$ . All these manifest that we need distributed learning to relieve stress from computation time and memory cost.

Before introducing our method, some additional notation is needed. Let  $\Psi = (\Psi_1, \dots, \Psi_p)$  denote  $n \times pd_n$  design matrix of the B-spline bases and  $\beta = (\beta_1^T, \beta_2^T, \dots, \beta_p^T)^T$  be the length- $pd_n$  coefficient vector. If  $A \subset \{1, \dots, p\}$ , we denote the  $n \times d_n|A|$  submatrix of  $\Psi$  by  $\Psi_A$  where for each  $j \in A$ ,  $\Psi_j$  represents the submatrix in the corresponding order. Correspondingly,  $\beta_A$  is the coefficients of  $\Psi_A$ . For parallel computing, assume  $X$  has been column-wisely partitioned into  $m$  groups, where  $m$  represents a pre-specified number of local machines one can access. If  $X_j$  is assigned to the  $i$ th group and it is the  $k$ th predictor in group  $i$ , we denote it by  $X_k^{(i)}$ . Note that there is a one-to-one mapping between the original predictor index  $j$  and the  $(i, k)$  pair. For convenience, define the mapping from the  $(i, k)$  pair to the original index as  $j = \zeta(i, k)$ . We denote the  $i$ th part of  $X$  by  $X^{(i)} = (X_1^{(i)}, X_2^{(i)}, \dots, X_{p_i}^{(i)})$  which are stored in local machine  $i$ ,  $i = 1, \dots, m$  and its spline basis matrix is denoted by  $\Psi^{(i)}$ . Excluding  $\Psi^{(i)}$ , we denote the remaining submatrix of  $\Psi$  by  $\Psi^{(-i)}$ . Let  $S^{(i)}$  denote the true set of important variables in the  $i$ th group, that is,  $S^{(i)} = \{\zeta(i, k) : f_k^{(i)} \neq 0\}$ , with  $s_i = |S^{(i)}|$ , and let  $S^{c(i)} = \{\zeta(i, k) : f_k^{(i)} = 0\}$  denote its complement. Thus,  $S$  is the union of  $S^{(i)}$ ,  $i = 1, \dots, m$ , and  $S^c$  denotes its complement.  $\Psi_S^{(i)}$  is the submatrix of  $\Psi^{(i)}$  consisting of spline basis of important predictors in the  $i$ th group and  $\Psi_{S^c}^{(i)}$  is the basis matrix for the noise predictors.

### 3. DDAC-SpAM Algorithm

Since  $X$  has already been column-wisely partitioned, each local machine stores one subset of the predictors and  $Y$ . Before the parallel variable selection, let us begin with a decorrelation step for the additive model.

Reformulated as the linear combination of basis functions, we have

$$Y = \Psi\beta + Z + \varepsilon, \quad (4)$$

where  $Z = (z_1, \dots, z_n)^T$  with  $z_i = \sum_{j=1}^p [f_j(x_{ij}) - f_{nj}(x_{ij})]$ ,  $i = 1, \dots, n$ .

The most intuitive way to reduce correlation is orthogonalizing the basis matrix  $\Psi$  to make its columns uncorrelated by left-multiplication. If  $\Psi$  has full column rank with  $n > pd_n$ , we write  $\Psi$  via singular value decomposition as  $\Psi = UDV^T$ , where  $U$  is a  $n \times pd_n$  tall matrix with orthonormal columns,  $D$  is a  $pd_n \times pd_n$  diagonal matrix and  $V$  is a  $pd_n \times pd_n$  orthogonal matrix. Then, we set  $\tilde{\Psi} = F\Psi$ . Here,  $F = UD^{-1}U^T$ . It is easy to see that the columns of  $\tilde{\Psi}$  are orthogonal. Actually,  $F$  can be calculated in the central machine by

$$\left( \sum_{i=1}^m \Psi^{(i)} \Psi^{(i)T} \right)^{\frac{1}{2}},$$

where the  $n \times n$  matrix  $\Psi^{(i)} \Psi^{(i)T}$  is transmitted from local machine and  $A^+$  denotes the Penrose-Moore pseudo-inverse of  $A$ .

Left multiplying  $F$  on both sides of (4), we get

$$FY = F\Psi\beta + FZ + F\varepsilon.$$

It can be denoted as

$$\tilde{Y} = \tilde{\Psi}\beta + \tilde{Z} + \tilde{\varepsilon}. \tag{5}$$

The spline basis matrix  $\tilde{\Psi}$  satisfies  $\tilde{\Psi}_i^T \tilde{\Psi}_j = 0$  for any  $i \neq j$  and  $\tilde{\Psi}_i^T \tilde{\Psi}_i = I_{d_n}$ .

The group lasso working mechanism for the  $i$ th data subgroup  $\{\tilde{Y}, \tilde{\Psi}^{(i)}\}$  can be shown as follow. First, the optimization object (3) would be

$$L(\beta^{(i)}) = \left\| \tilde{Y} - \sum_{k=1}^{p_i} \tilde{\Psi}_k^{(i)} \beta_k^{(i)} \right\|^2 + \lambda_n \sum_{k=1}^{p_i} \frac{1}{\sqrt{n}} \|\tilde{\Psi}_k^{(i)} \beta_k^{(i)}\|. \tag{6}$$

As shown in Yuan and Lin (2006) and Ravikumar et al. (2009), a solution to (6) satisfies

$$\hat{\beta}_k^{(i)} = \left[ 1 - \frac{\lambda_n}{\|P_k^{(i)}\|} \right]_+ P_k^{(i)},$$

where  $P_k^{(i)} = \tilde{\Psi}_k^{(i)T} \tilde{Y}$ .

Combining with (5), we can derive that

$$\begin{aligned} P_k^{(i)} &= \tilde{\Psi}_k^{(i)T} (\tilde{\Psi}_k^{(i)} \beta_k^{(i)} + \tilde{\Psi}_{-k}^{(i)} \beta_{-k}^{(i)} + \tilde{\Psi}^{(-i)} \beta^{(-i)} + \tilde{Z} + \tilde{\varepsilon}) \\ &= \beta_k^{(i)} + \tilde{\Psi}_k^{(i)T} \tilde{\Psi}_{-k}^{(i)} \beta_{-k}^{(i)} + \tilde{\Psi}_k^{(i)T} \tilde{\Psi}^{(-i)} \beta^{(-i)} + \tilde{\Psi}_k^{(i)T} \tilde{Z} + \tilde{\Psi}_k^{(i)T} \tilde{\varepsilon} \\ &= \beta_k^{(i)} + \tilde{\Psi}_k^{(i)T} \tilde{Z} + \tilde{\Psi}_k^{(i)T} \tilde{\varepsilon} \end{aligned} \tag{7}$$

where  $\tilde{\Psi}_{-k}^{(i)} = (\tilde{\Psi}_1^{(i)}, \dots, \tilde{\Psi}_{k-1}^{(i)}, \tilde{\Psi}_{k+1}^{(i)}, \dots, \tilde{\Psi}_{p_i}^{(i)})$  and  $\beta_{-k}^{(i)} = (\beta_1^{(i)T}, \dots, \beta_{k-1}^{(i)T}, \beta_{k+1}^{(i)T}, \dots, \beta_{p_i}^{(i)T})^T$ . Since the last two terms of (7) can be bounded by Condition 1, with a mild condition for  $\Psi\Psi^T$  to be presented in Section 5,  $P_k^{(i)}$  converges to  $\beta_k^{(i)}$  almost at the same rate as that with the full data.

When  $pd_n \geq n$ , SVD of  $\Psi$  generates a  $n \times n$  orthogonal matrix  $U$ , a  $pd_n \times n$  matrix  $V$  with only orthonormal columns and  $D$  is a  $n \times n$  diagonal matrix. Then  $F$  becomes  $(\sum_{i=1}^m \Psi^{(i)} \Psi^{(i)T})^{-\frac{1}{2}}$ . Although the columns of  $\tilde{\Psi}$  are not exactly mutually orthogonal, that is, for some  $i \neq j$ ,  $\tilde{\Psi}_i^T \tilde{\Psi}_j \neq 0$ , according to Khatri and Pillai (1965), we have  $E(\tilde{\Psi}^T \tilde{\Psi}) = (n/pd_n)I_{pd_n}$ , which means that any two columns of  $\tilde{\Psi}$  are orthogonal in expectation. Thus, we can still apply the same decorrelation step to get a new response  $\tilde{Y}$  and the design matrix  $\tilde{\Psi}$ .

The decorrelation operation mainly aims to lower the correlation between the basis functions in different blocks. To show it visually, we now present a simple example with  $n < pd_n$ . In particular, we sample  $X$  from zero mean normal distribution with covariance matrix  $\Sigma = [\sigma_{ij}]$ , where  $\sigma_{ii} = 1$ ,  $\sigma_{ij} = \rho$  with  $i \neq j$  and  $(n, p) = (500, 1000)$ . A cubic B-spline with  $d_n = 5$  is used. We focus on comparing  $\tilde{\rho}_{ij} := \text{tr}(\tilde{\Psi}_i^T \tilde{\Psi}_j) / (\|\tilde{\Psi}_i\|_F \|\tilde{\Psi}_j\|_F)$  and  $\rho_{ij} := \text{tr}(\Psi_i^T \Psi_j) / (\|\Psi_i\|_F \|\Psi_j\|_F)$ ,  $1 \leq i < j \leq p$ . The difference between these two terms is affected by the dependence between basis functions of different covariates. We call them quasi-correlation here. Figure 1 shows the boxplots of quasi-correlation before and after the decorrelation step when  $\rho$  increases. It can be seen that while  $\rho_{ij}$  increases with  $\rho$ ,  $\tilde{\rho}_{ij}$  is stable throughout the range of  $\rho$  at a very low level, which means the decorrelation step reduces correlation between additive components in high-dimensional additive model significantly.

After the decorrelation step, since  $\{\tilde{\Psi}_k^{(i)}, k = 1, \dots, p_i\}$  is not exactly column-orthogonal, we cannot apply the SpAM backfitting algorithm (Ravikumar et al. 2009) directly with  $\{\tilde{Y}, \tilde{\Psi}^{(i)}\}$  on

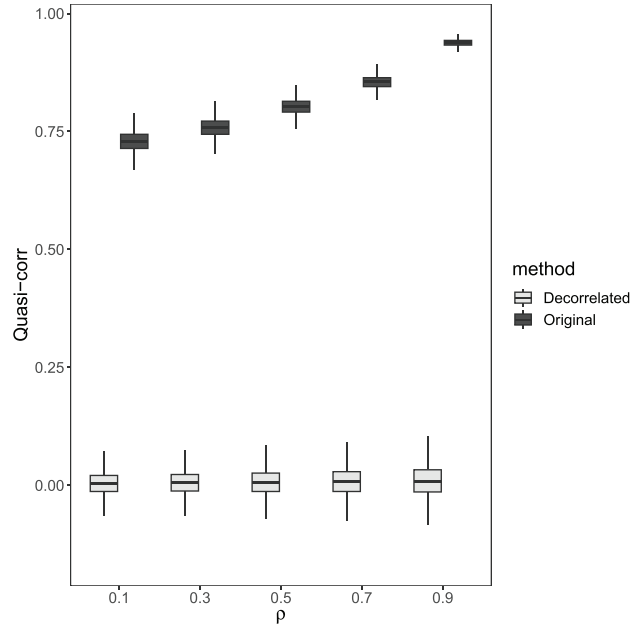


Figure 1. Comparison of quasi-correlations for  $\Psi$  and  $\tilde{\Psi}$ .

the  $i$ th local machine. Following Simon and Tibshirani (2012), we use a method similar to the standardized group lasso to solve this problem.

Specifically, we apply QR decomposition for each block  $\tilde{\Psi}_k^{(i)}$ ,  $i = 1, \dots, m$ ,  $k = 1, \dots, p_i$  into the product of an orthogonal matrix  $\tilde{Q}_k^{(i)}$  and an upper triangular matrix  $\tilde{R}_k^{(i)}$ . Then, the  $i$ th local machine runs the SpAM backfitting algorithm with  $\{\tilde{Y}, \tilde{Q}^{(i)}\}$  to solve the following problem

$$\begin{aligned} \hat{\theta}^{(i)} &= \arg \min_{\theta^{(i)} = (\theta_1^{(i)T}, \dots, \theta_{p_i}^{(i)T})^T} \frac{1}{n} \|\tilde{Y} - \tilde{Q}^{(i)} \theta^{(i)}\|^2 + \lambda_n \sum_{k=1}^{p_i} \|\tilde{Q}_k^{(i)} \theta_k^{(i)}\|, \\ i &= 1, \dots, m \end{aligned} \tag{8}$$

and select variables, where  $\tilde{Q}^{(i)} = (\tilde{Q}_1^{(i)}, \dots, \tilde{Q}_{p_i}^{(i)})$ .

The original coordinates  $\hat{\beta}^{(i)}$  can be back-solved by

$$\hat{\beta}_k^{(i)} = (\tilde{R}_k^{(i)})^{-1} \hat{\theta}_k^{(i)}. \tag{9}$$

However, this is unnecessary for the purpose of feature selection since  $\hat{\theta}_k^{(i)} = 0$  implies  $\hat{\beta}_k^{(i)} = 0$ . The local machines only need to transfer the selected important variables and their basis functions to the central machine. The final estimates of  $\beta^{(i)}$  and  $f_j(X_j)$  will be computed on the central machine.

Let  $\hat{S}^{(i)} = \{\zeta(i, k) : \hat{\theta}_k^{(i)} \neq 0\}$  denote the  $i$ th estimated set of important variables, for  $i = 1, \dots, m$ , and  $\hat{S}$  be their union. The details of DDAC-SpAM are provided in Algorithm 1.

**Remark 1.** Steps 1 and 2 are used for data initialization and division. Steps 3–5 are the decorrelation steps. Step 6 and 7 are distributed feature selection and final refinement steps, respectively. In Step 4, we use  $\sum_{i=1}^m \Psi^{(i)} \Psi^{(i)T} + rI_n$  instead of  $\sum_{i=1}^m \Psi^{(i)} \Psi^{(i)T}$  for robustness. Besides, using ridge regression in Step 7 instead of ordinary least squares is also for robustness.

Now, we analyze the computational complexity and memory consumption of DDAC-SpAM. For convenience, we assume that



---

**Algorithm 1:** Divide, decorrelate and conquer SpAM (DDAC-SpAM)

---

- Input:**  $Y, X, d_n$ , the number of machines  $m$ , the ridge regularization parameter  $r$ .
- 1 On the central machine, store and standardize  $Y$  to get  $\underline{Y}$ ;
  - 2 Randomly divide predictors into  $m$  parts:  $X^{(1)}, \dots, X^{(m)}$  and allocate  $(\underline{Y}, X^{(i)})$  to the  $i$ th local machine for  $i = 1, \dots, m$ ;
  - 3 On the  $i$ th local machine,  $i = 1, \dots, m$ , generate spline basis matrix  $\Psi^{(i)}$  for  $X^{(i)}$ , standardize every column of  $\Psi^{(i)}$  to get  $\underline{\Psi}^{(i)}$  and transmit  $\underline{\Psi}^{(i)} \underline{\Psi}^{(i)T}$  to the central machine;
  - 4 On the central machine, compute  $F = (\sum_{i=1}^m \underline{\Psi}^{(i)} \underline{\Psi}^{(i)T} + rI_n)^{-1/2}$  and transmit it to the local machines;
  - 5 On the  $i$ th local machine,  $i = 1, \dots, m$ , compute  $\tilde{\Psi}^{(i)} = F \underline{\Psi}^{(i)}$  and  $\tilde{Y} = F \underline{Y}$ ;
  - 6 On the  $i$ th local machine,  $i = 1, \dots, m$ , (a) perform the QR factorization  $\tilde{\Psi}_k^{(i)} = \tilde{Q}_k^{(i)} \tilde{R}_k^{(i)}$ , for  $k = 1, \dots, p_i$ ; (b) run the SpAM backfitting algorithm to solve (8); (c) push  $\hat{S}^{(i)}$  and  $\Psi_{\hat{S}^{(i)}}$  to the central machine;
  - 7 On the central machine, combine  $\Psi_{\hat{S}^{(i)}}$ ,  $i = 1, \dots, m$ , to get  $\Psi_{\hat{S}}$ . Apply ridge regression on  $(Y, \Psi_{\hat{S}})$  and get  $\hat{\beta}_{\hat{S}}$ ;
- Output:**  $\hat{S}$  and  $\hat{\beta}_j, j \in \hat{S}$ .
- 

the  $p$  features are evenly distributed to  $m$  parts. Excluding spline interpolation, the costs of the decorrelation operation and QR factorization are  $O(n^3 + n^2pd_n/m + n^2m)$  and  $O(npd_n^2/m)$  per local machine, respectively, so the total cost is  $O(n^3 + n^2pd_n/m + n^2m)$ . For parallel estimation, within each iteration of the SpAM backfitting algorithm,  $O(npd_n/m)$  calculations are required. Assume the number of loops is  $k$ . So the total computational cost is  $O(n^3 + n^2pd_n/m + n^2m + knpd_n/m)$  for DDAC-SpAM, compared with  $O(knpd_n + npd_n^2)$  for SpAM on a single machine. Meanwhile, the memory consumption of every local machine through the entire algorithm is decreased roughly by a factor of  $m$ . As shown above, DDAC-SpAM can significantly speed up computation and relax memory requirements. This will be demonstrated in the numerical study section.

#### 4. Statistical Inference via Debiased DDAC-SpAM

In the previous section, we have developed the DDAC-SpAM algorithm for feature selection of the sparse additive model. With the selected variables, we further apply ridge regression to obtain an estimation of the coefficients  $\beta$  in (4). In this section, we study the statistical inference for  $\beta$ .

Naturally associated with the sparse additive model  $y_i = \sum_{j=1}^p f_j(x_{ij}) + \varepsilon_i$ , we focus on the fundamental hypothesis testing problem  $H_0: f_j = 0$  versus  $H_1: f_j \neq 0$  for some  $1 \leq j \leq p$ . In terms of the B-spline basis expansion under our distributed computing setting, this is equivalent to

$$H_0: \beta_k^{(i)} = 0 \text{ versus } H_1: \beta_k^{(i)} \neq 0 \quad (10)$$

for  $1 \leq i \leq m$  and  $1 \leq k \leq p_i$ .

Based on recent developments in high-dimensional inference for linear models (Van de Geer et al. 2014; Cai, Zhang, and Zhou 2022), we construct the following debiased DDAC-SpAM estimator

$$\hat{\beta}^u = \hat{\beta} + \frac{pd_n}{n} \tilde{\Psi}^T (\tilde{Y} - \tilde{\Psi} \hat{\beta}), \quad (11)$$

where  $\hat{\beta}$  is the DDAC-SpAM estimate of  $\beta$  obtained from (9). Restricting ourselves to the  $k$ th variable in the  $i$ th machine,  $\hat{\beta}_k^{u(i)}$  enjoys the property that its scaled and decorrelated version  $\hat{M}_k^{(i)} (\hat{\beta}_k^{u(i)} - \beta_k^{(i)})$  approximately follows a  $d_n$ -dimensional standard normal distribution, where

$$\hat{M}_k^{(i)} = (\tilde{\Psi}_k^{(i)T} F F^T \tilde{\Psi}_k^{(i)})^{-1/2} n / (pd_n \hat{\sigma}), \quad (12)$$

in which  $\hat{\sigma}$  is an estimate for  $\sigma$  (Theorem 2). Therefore, we define our test statistic  $\mathcal{T}_k^{(i)} := \|\hat{M}_k^{(i)} \hat{\beta}_k^{u(i)}\|^2$ , which follows  $\chi^2(d_n)$  asymptotically under  $H_0$ . The notation  $\mathcal{T}_j := \mathcal{T}_k^{(i)}$  is also used when  $j = \zeta(i, k)$ . With significance level  $\alpha_0$ , we reject the null hypothesis of (10) when  $\mathcal{T}_k^{(i)} > F_{d_n}^{-1}(1 - \alpha_0)$ , where  $F_{d_n}$  is the cumulative distribution function for  $\chi^2(d_n)$ . The detailed testing procedure is presented in Algorithm 2.

*Remark 2.* In Cai, Zhang, and Zhou (2022) and Van de Geer et al. (2014), the debiased estimator is constructed as  $\hat{\beta}^u = \hat{\beta} + \hat{\Theta} \Psi^T (Y - \Psi \hat{\beta}) / n$ , where  $\hat{\Theta}$  is an estimate of  $(\Psi^T \Psi / n)^{-1}$ . When working with decorrelated  $\tilde{\Psi}$  and  $\tilde{Y}$ , we have  $(\tilde{\Psi}^T \tilde{\Psi} / n)^{-1} \approx pd_n I_{pd_n}$  (Khatri and Pillai 1965), which leads to the definition in (11).

---

**Algorithm 2:** Inference via Debiased DDAC-SpAM ( $H_0: \beta_{k_0}^{(i_0)} = 0$  versus  $H_1: \beta_{k_0}^{(i_0)} \neq 0$ )

---

- Input:**  $Y, X, d_n$ , the number of machines  $m$ , the ridge regularization parameter  $r$ , the significance level  $\alpha_0$ .
- 1 On the central machine, store and center  $Y$  to get  $\underline{Y}$ ;
  - 2 Steps 2–5 of Algorithm 1;
  - 3 On the  $i$ th local machine,  $i = 1, \dots, m$ , perform the QR factorization  $\tilde{\Psi}_k^{(i)} = \tilde{Q}_k^{(i)} \tilde{R}_k^{(i)}$ , for  $k = 1, \dots, p_i$ , run the SpAM backfitting algorithm to solve (8), compute  $\hat{\beta}^{(i)}$  by  $\hat{\beta}_k^{(i)} = (\tilde{R}_k^{(i)})^{-1} \hat{\beta}_k^{(i)}$ , and push  $\hat{Y}^{(i)} = \underline{\Psi}^{(i)} \hat{\beta}^{(i)}$  to the central machine;
  - 4 On the central machine, fetch  $\hat{\beta}_{k_0}^{(i_0)}$  and  $\tilde{\Psi}_{k_0}^{(i_0)}$  from the  $i_0$ th machine, compute  $\hat{\varepsilon} = Y - \sum_{i=1}^m \hat{Y}^{(i)}$ ,  $\hat{\sigma} = \|\hat{\varepsilon}\| / \sqrt{n}$ ,  $\hat{M}_{k_0}^{(i_0)} = (\tilde{\Psi}_{k_0}^{(i_0)T} F F^T \tilde{\Psi}_{k_0}^{(i_0)})^{-1/2} n / (pd_n \hat{\sigma})$ ,  $\hat{\beta}_{k_0}^{u(i_0)} = \hat{\beta}_{k_0}^{(i_0)} + pd_n \tilde{\Psi}_{k_0}^{(i_0)T} F \hat{\varepsilon} / n$  and  $\mathcal{T}_{k_0}^{(i_0)} = \|\hat{M}_{k_0}^{(i_0)} \hat{\beta}_{k_0}^{u(i_0)}\|^2$ ;
- Output:** “Reject” if  $\mathcal{T}_{k_0}^{(i_0)} > F_{d_n}^{-1}(1 - \alpha_0)$ ; “Accept” otherwise.
- 

#### 5. Theoretical Results

In this section, we provide the theoretical framework for DDAC-SpAM to show it is variable selection consistent (*sparsistent*)

under mild conditions. On top of that, we further derive the asymptotic distribution for the debiased DDAC-SpAM estimator.

For results in this section, we will treat  $X$  as random. When  $pd_n \geq n$ , let the spline interpolation  $\Psi = UDV^T$ , where  $U$  is a  $n \times n$  orthogonal,  $D$  is a  $n \times n$  diagonal matrix and  $V$  satisfies  $V^T V = I_n$ .  $\tilde{\Psi} = F\Psi = UV^T$  satisfies  $\tilde{\Psi}\tilde{\Psi}^T = I_n$ . All  $n \times pd_n$  matrices whose rows are orthonormal (e.g.,  $\tilde{\Psi}$ ) form *Stiefel manifold*  $\mathcal{V}(n, pd_n)$  (Downs 1972).

For clarity, we review the definition of uniform distribution for a random matrix.

**Definition 1 (Chikuse 2003).** A random  $n \times p$  matrix  $H$  is uniformly distributed on  $\mathcal{V}(n, p)$ , written  $H \sim \text{Uniform}(\mathcal{V}(n, p))$ , if  $H$  has the same distribution as  $HO$  for any fixed  $p \times p$  orthogonal matrix  $O$ .

Besides **Condition 1**, we further make the following assumptions for  $\Psi$ .

**Condition 2.**  $\tilde{\Psi} \sim \text{Uniform}(\mathcal{V}(n, pd_n))$ .

We allow the minimum eigenvalue of  $\Psi\Psi^T$  decay with sample size at a certain rate.

**Condition 3.**  $P(\lambda_{\min}(\Psi\Psi^T/pd_n) > \delta n^{\alpha-1}) \geq 1 - \exp(-\xi n^\gamma)$ , for some  $0 < \alpha \leq 1$  and  $\delta, \xi, \gamma > 0$ .

Similar conditions to **Condition 2** have been imposed on the design matrix of linear model (Jia and Rohe 2015). **Condition 3** is related to **Theorem 2** in Ravikumar et al. (2009) in which they require the eigenvalues of  $n^{-1}\Psi^T\Psi$  to be bounded by constants. When the number of covariates diverges with  $n$ , **Condition 3** is more general than theirs.

We also assume that the truncation size  $d_n$ , regularization parameter  $\lambda_n$  and the number of important variables  $s$  satisfy

**Condition 4.**  $d_n \rightarrow \infty, \tilde{\lambda}_n \rightarrow 0, \tilde{\lambda}_n^{-2}n^{-1}s^2d_n \rightarrow 0, \tilde{\lambda}_n^{-2}n^{1-\alpha}s^2d_n^3 \rightarrow 0, \tilde{\lambda}_n^{-2}n^{-\alpha}sd_n \rightarrow 0$  and  $\sqrt{s}\tilde{\lambda}_n/\rho_n \rightarrow 0$ , where  $\tilde{\lambda}_n = \sqrt{pd_n}\lambda_n$  and  $\rho_n = \min_{j \in S} \|\beta_j\|_\infty$ .

Similar conditions were assumed in many high-dimensional additive model variable selection literatures, such as condition (B2) for Theorem 3 in Huang, Horowitz, and Wei (2010).

Additionally, assume

**Condition 5.**  $p = o(\exp(sd_n))$ .

To clarify the implications of **Conditions 4** and **5**, assume the number of important variables is bounded, that is,  $s = O(1)$ . Then, in practice, we can set  $d_n \asymp n^{1/5}$ . The order of dimension  $p_n$  can be as large as  $o(\exp(n^{1/5}))$ . If  $1/\rho_n = o(n^{\alpha/2-1/5}/\log n)$ , a suitable choice for the regularization parameter  $\tilde{\lambda}_n$  would be  $n^{1/5-\alpha/2} \log n$  for some  $2/5 < \alpha \leq 1$ .

If we impose a slightly more strict requirement on  $\alpha$  (e.g.,  $3/5 < \alpha \leq 1$ ), then the sparsity  $s$  is allowed to increase with  $n$ . For example, assuming  $s \asymp n^{1/10}, p = o(\exp(n^{3/10}))$ ,  $d_n \asymp n^{1/5}$  and  $1/\rho_n = o(n^{\alpha/2-7/20}/\log n)$ , we can choose the regularization parameter  $\tilde{\lambda}_n \asymp n^{3/10-\alpha/2} \log n$  so that both **Conditions 4** and **5** are satisfied.

The key of our algorithm is to reduce the correlation between predictors which leads to a milder constraint for the correlation structure of variables or  $f_j(X_j)$  within **Theorem 1** than before. It can be reflected in two aspects. First, there is no assumption for the correlation between important variables that are distributed to different local machines, since the bound of this kind of correlation is reduced to

$$P \left\{ \left\| \frac{pd_n \tilde{\Psi}_S^{(i)T} \tilde{\Psi}_S^{(-i)}}{n} \right\| \leq \frac{C_1 \tilde{\lambda}_n}{\sqrt{s}} \right\} \rightarrow 1,$$

for some  $C_1 > 0$ , as shown in Lemma S.2 in the supplementary material of this article. Second, there is no assumption for the correlation between important variables and irrelevant variables. Specifically, we do not need a version of *irrepresentable condition* (Zhao and Yu 2006) for selection consistency of DDAC-SpAM. In previous works, such as Ravikumar et al. (2009), this kind of condition can be formulated as an upper bound for

$$\max_{j \in S^c} \left\| \left( \frac{1}{n} \Psi_j^T \Psi_S \right) \left( \frac{1}{n} \Psi_S^T \Psi_S \right)^{-1} \right\|,$$

while this bound exists in our work by

$$P \left\{ \max_{j \in S^c(i)} \left\| \frac{pd_n \tilde{\Psi}_S^{(i)T} \tilde{\Psi}_k^{(i)}}{n} \right\| \leq \frac{C_2}{\sqrt{s}}, \left\| \left( \frac{pd_n \tilde{\Psi}_S^{(i)T} \tilde{\Psi}_S^{(i)}}{n} \right)^{-1} \right\| \leq C_3 \right\} \rightarrow 1,$$

for some  $C_2, C_3 > 0$ . The details and proof of these results can be found in Lemmas S.2, S.3, S.4 of the supplementary material.

**Theorem 1.** Assuming **Conditions 1–5** hold and  $s_i > 0$ , the following inequality holds for sufficiently large  $n$ :

$$P(\hat{S}^{(i)} = S^{(i)}) \geq 1 - \exp(-\xi n^\gamma) - 16(p_i - s_i + 1) \exp(-sd_n) \rightarrow 1,$$

that is, the local estimator on machine  $i$  is sparsistent.

The convergence rate of  $P(\hat{S}^{(i)} = S^{(i)})$  is controlled by two  $o(1)$  terms, where  $\exp(-\xi n^\gamma)$  directly follows from **Condition 3**, and  $16(p_i - s_i + 1) \exp(-sd_n)$ , as a combined rate, has connections to both the Gaussian assumption of  $\varepsilon$  and the uniform assumption in **Condition 2**. The additional technical condition  $s_i > 0$  is not critical in our numerical studies, as we will show in the following sections. The proof of **Theorem 1** is provided in the supplementary material.

After aggregating the results from local machines, we can derive the following corollary.

**Corollary 1.** Under **Conditions 1–5**, the central estimator is sparsistent:

$$P(\hat{S} = S) \geq 1 - \exp(-\xi n^\gamma) - 16(p - s + 1) \exp(-sd_n) \rightarrow 1.$$

Next, we will establish the theoretical foundation of **Algorithm 2**. To ensure our test statistic  $\mathcal{T}$  follow the desired Chi-squared distribution asymptotically, we need two additional conditions.

**Condition 6.**  $P(\lambda_{\max}(\Psi\Psi^T/pd_n) \leq \delta') \geq 1 - \exp(-\xi' n^{\gamma'})$  for some  $\delta', \xi', \gamma' > 0$ .

**Condition 7.**  $ns^2d_n^{-3} \rightarrow 0$  and  $\sqrt{s}\tilde{\lambda}_n n^{1/4} \rightarrow 0$ .

**Condition 6** resembles **Condition 3** and requires the eigenvalues of  $\Psi\Psi^T$  to be bounded from above with high probability. **Condition 7** strengthens **Condition 4** to make the more challenging statistical inference a feasible task. One immediate implication of **Condition 7** is that more knots are needed for the B-spline basis. With  $s = O(1)$ , it is necessary that  $d_n \gg n^{1/3}$ . For instance, we may set  $d_n \asymp n^{1/3} \log n$ , which along with  $\tilde{\lambda}_n \asymp n^{1/6-\alpha/2} \log n$ ,  $1/\rho_n = o(n^{\alpha/2-1/6}/\log n)$  and  $p = o(\exp(n^{1/3} \log n))$  would satisfy all requirements of **Conditions 4, 5, and 7** for  $5/6 < \alpha \leq 1$ .

**Theorem 2.** Assuming **Conditions 1–7** hold and  $s_i > 0$ , the debiased DDAC-SpAM estimator  $\hat{\beta}^u$  has the following asymptotic distribution:

$$\widehat{M}_k^{(i)} \left( \hat{\beta}_k^{u(i)} - \beta_k^{(i)} \right) \xrightarrow{d} \mathcal{N}(0, I_{d_n}),$$

for all  $1 \leq i \leq m$  and  $1 \leq k \leq p_i$ , where  $\widehat{M}_k^{(i)} = (\widetilde{\Psi}_k^{(i)T} FF^T \widetilde{\Psi}_k^{(i)})^{-1/2} n / (pd_n \sigma)$  is (12) with  $\hat{\sigma}$  replaced by its true value  $\sigma$ . In addition, the test statistic

$$\mathcal{T}_k^{(i)} = \|\widehat{M}_k^{(i)} \hat{\beta}_k^{u(i)}\|^2 \xrightarrow{d} \chi^2(d_n)$$

when  $\beta_k^{(i)} = 0$ .

## 6. Simulation Studies

### 6.1. Performance Comparison Under Different Correlation Structures and Distributions

To study the performance of the DDAC-SpAM procedure on simulated datasets, we divide feature space evenly and randomly. Since the results are stable for different choices of  $r$ , we fix it to be 1 throughout the numerical studies for simplicity. For each  $f_j$ , we use a cubic B-spline parameterization with  $d_n = \lceil 0.1n^{1/3} \log n \rceil$  according to the discussions after **Condition 7**. For comparison, we include the full data SpAM with a ridge refinement (SpAM) and SpAM with separated feature space without decorrelation (DAC-SpAM). The oracle method that employs Step 7 of **Algorithm 1** with  $\Psi_{\hat{\zeta}}$  replaced by  $\Psi_S$  is also included as the benchmark. We report the false positives (FP), false negatives (FN), the mean squared error  $\|\hat{h} - h\|^2$  (MSE) and computational time (Time). We use `gglasso` (Yang and Zou 2017) and `glmnet` (Friedman, Hastie, and Tibshirani 2010) with 5-fold cross-validation to fit group lasso and ridge regression, respectively. We consider an exponentially-decay sequence for  $\lambda_n$ , whose value varies from  $\lambda_n^{(1)} > \lambda_n^{(2)} > \dots > \lambda_n^{(500)}$ , where  $\lambda_n^{(1)}$  is the smallest  $\lambda$  value such that all coefficient estimates are zero and  $\lambda_n^{(500)} = 0.001\lambda_n^{(1)}$ .

We define the signal-to-noise ratio

$$\text{SNR} = \frac{\text{var}(h(X))}{\text{var}(\varepsilon)}.$$

**Independent Predictors.** We first consider the case with independent predictors. Two examples where predictors follow a uniform distribution and normal distribution are analyzed, respectively.

**Example 1 (SNR  $\approx 15$ ).** Following **Example 1** in Meier, Van de Geer, and Bühlmann (2009), we generate the data from the following additive model:

$$y_i = 2g_1(x_{i1}) + 1.6g_2(x_{i2}) - 4g_3(x_{i3}, 2) + g_4(x_{i4}) + 1.5\varepsilon_i,$$

with

$$g_1(x) = x, \quad g_2(x) = x^2 - \frac{25}{12}, \quad g_3(x, \omega) = \sin(\omega x), \\ g_4(x) = e^{-x} - 2/5 \cdot \sinh(5/2),$$

where the covariates  $X = (X_1, \dots, X_p)$  are simulated from independent Uniform  $(-2.5, 2.5)$  and  $\varepsilon_i \stackrel{\text{iid}}{\sim} \mathcal{N}(0, 1)$ .

**Example 2 (SNR  $\approx 15$ ).** In this example, the covariates  $X = (X_1, \dots, X_p)$  are simulated from independent standard normal distribution. The model is

$$y_i = 5g_1(x_{i1}) + 2.1g_5(x_{i2}) + 13.2g_6(x_{i3}, \frac{\pi}{4}) + 17.2g_7(x_{i4}, \frac{\pi}{4}) + 2.56\varepsilon_i,$$

with

$$g_5(x) = (x - 1)^2, \quad g_6(x, \omega) = \frac{\sin(\omega x)}{2 - \sin(\omega x)}, \\ g_7(x, \omega) = 0.1 \sin(\omega x) + 0.2 \cos(\omega x) + 0.3 \sin^2(\omega x) \\ + 0.4 \cos^3(\omega x) + 0.5 \sin^3(\omega x),$$

and  $\varepsilon_i \stackrel{\text{iid}}{\sim} \mathcal{N}(0, 1)$ .

**Dependent Predictors.** For dependent predictors with different distributions, we investigate three different correlation structures.

**Example 3 (SNR  $\approx 6.7$ ).** Following **Example 3** in Meier, Van de Geer, and Bühlmann (2009), the covariates are generated with the following random-effects model:

$$X_j = \frac{W_j + tU_{\lfloor j/20 \rfloor}}{1 + t}, j = 1, \dots, p,$$

where  $W_1, \dots, W_p, U_1, \dots, U_{\lfloor p/20 \rfloor} \stackrel{\text{iid}}{\sim}$  Uniform  $(0, 1)$ . By construction, the  $p$  predictors are partitioned into segments of size 20. Variables in different segments are independent while the variables in each segment are dependent through the shared  $U$  variable. As a result, the correlation strength within each segment is controlled by  $t$ . Here, we set  $t = 1.5$ , leading to a correlation between  $X_i$  and  $X_j$  to be 0.6 when they are in the same segment. The model is

$$y_i = 2.5g_1(x_{i1}) + 2.6g_5(x_{i2}) + g_6(x_{i3}, 2\pi) + g_7(x_{i4}, 2\pi) + 0.3\varepsilon_i,$$

where  $\varepsilon_i \stackrel{\text{iid}}{\sim} \mathcal{N}(0, 1)$ .

**Example 4 (SNR  $\approx 6.7$ ).** The setting is the same as **Example 3** except that  $X_j = (W_j + tU)/(1 + t)$  and  $U \sim$  Uniform  $(0, 1)$ . We set  $t = 1.5$  leading to the pairwise correlation of all covariates being 0.6.



**Example 5 (SNR  $\approx 18$ ).** The covariates are generated according to a multivariate normal distribution with zero mean and covariance matrix  $\Sigma = [\sigma_{ij}]$ , where  $\sigma_{ii} = 1$  and  $\sigma_{ij} = 0.5$  for  $i \neq j$ .  $Y$  is generated with

$$y_i = 2.5g_1(x_{i1}) + g_5(x_{i2}) + 6.5g_6(x_{i3}, \frac{\pi}{4}) + 8.5g_7(x_{i4}, \frac{\pi}{4}) + 1.2\varepsilon_i,$$

where  $\varepsilon_i \stackrel{\text{iid}}{\sim} \mathcal{N}(0, 1)$ .

For all examples, the feature dimension and the sample size are fixed at  $p = 10,000$  and  $n = 500$ , respectively, which leads to  $d_n = 5$ . The number of machines is fixed as  $m = 20$ . We run each simulation for 100 times and report the average performance in Table 1.

Several conclusions can be drawn from Table 1. In Examples 1 and 2 where all variables are independent, DDAC-SpAM and DAC-SpAM perform the best, closely mimicking the performance of Oracle in terms of MSE. The possible reason for the similar performance between DDAC-SpAM and DAC-SpAM is that the decorrelation step is not necessary under this independent setting. On the other hand, SpAM tends to select more irrelevant variables. This shows that for independent variables, distributed feature selection can enhance the selection accuracy. Also, we can see that both DDAC-SpAM and DAC-SpAM take much less time than SpAM, showing the power of distributed computing. In Examples 3–5 where the variables are dependent, the performances of both SpAM and DAC-SpAM deteriorate, possibly due to the violation of the irrepresentable condition. On the contrary, DDAC-SpAM is far less affected and achieves the overall best performance. In particular, it has much fewer false positives than the other two methods. This shows that the decorrelation step can handle such kinds of strong correlation structures. Also it leads to a smaller MSE than SpAM and DAC-SpAM for Examples 4 and 5. This shows the importance of the decorrelation step when there exists correlation among features. Similar to Examples 1 and 2, we observe DDAC-SpAM and DAC-SpAM are much faster to compute than SpAM. Interestingly, DDAC-SpAM takes significantly less time to compute

than DAC-SpAM, possibly due to a faster SpAM fitting after decorrelation.

## 6.2. Performance Comparison with Varying Number of Machines

Corollary 1 indicates the sparsistency property of DDAC-SpAM is irrelevant to the number of machines  $m$ . This is because in the ideal scenario where the decorrelation step produces perfectly independent covariates, the aggregated result from the local machines is identical to the full data estimator. In reality, correlation between variables can never be fully eliminated, in which case a large  $m$  value mainly has two effects. First, it increases bias by distributing correlated variables into different machines, which can potentially hurt the performance of DDAC-SpAM. Second, it tends to separate correlated important variables, encouraging their simultaneous selection.

In this experiment, we analyze the impact of the number of machines on the performance of DDAC-SpAM with simulated data. We fix the sample size  $n = 500$  and the dimension  $p = 10,000$ , and vary the number of machines  $m$  from 1 to 200 ( $m = 1, 10, 20, 100, 200$ ). Naturally, as  $m$  increases, each machine has a lower local dimension. The data is generated using Example 4 in Section 6.1.

We summarize the results in Figure 2. First, we observe that all three methods capture nearly all important variables. Compared with DAC-SpAM and SpAM, DDAC-SpAM has the smallest number of false positive variables and lowest estimation error. While DAC-SpAM suffers from high correlation between features when the computation is distributed, that is,  $m > 1$ , the feature selection and prediction performance of DDAC-SpAM is stable throughout the range of  $m$ . Besides, thanks to the distributed framework, the time consumption of DDAC-SpAM and DAC-SpAM decreases as  $m$  increases. Although decorrelation increases the computational complexity which is evident when the dataset is not partitioned, that is,  $m = 1$ , DDAC-SpAM takes less time than DAC-SpAM as  $m$  increases. The reason is that the reduced correlation between variables leads to less number of back-fitting loops required for convergence in the

**Table 1.** Average false positive (FP), false negative (FN), mean squared error (MSE), time (in seconds) over 100 repetitions and their standard deviations (in parentheses).

Model	Method	FP	FN	MSE	Time
Example 1	DDAC-SpAM	0.02 (0.14)	0.00 (0.00)	0.465 (0.09)	42.96 (2.55)
	DAC-SpAM	0.04 (0.20)	0.00 (0.00)	0.466 (0.09)	42.05 (2.97)
	SpAM	1.22 (3.55)	0.00 (0.00)	0.562 (0.20)	1524.47 (348.38)
	Oracle	NA (NA)	NA (NA)	0.464 (0.09)	NA (NA)
Example 2	DDAC-SpAM	0.03 (0.22)	0.00 (0.00)	2.369 (0.59)	42.60 (2.08)
	DAC-SpAM	0.01 (0.10)	0.00 (0.00)	2.365 (0.59)	41.70 (1.95)
	SpAM	0.61 (1.47)	0.00 (0.00)	2.496 (0.64)	1521.43 (243.08)
	Oracle	NA (NA)	NA (NA)	2.367 (0.59)	NA (NA)
Example 3	DDAC-SpAM	3.59 (3.58)	0.09 (0.29)	0.036 (0.03)	42.02 (2.18)
	DAC-SpAM	4.67 (3.15)	0.07 (0.26)	0.035 (0.02)	41.48 (2.20)
	SpAM	6.58 (6.39)	0.00 (0.00)	0.036 (0.01)	1542.65 (302.87)
	Oracle	NA (NA)	NA (NA)	0.026 (0.00)	NA (NA)
Example 4	DDAC-SpAM	0.01 (0.10)	0.03 (0.17)	0.030 (0.02)	43.44 (2.39)
	DAC-SpAM	41.23 (14.21)	0.00 (0.00)	0.055 (0.01)	76.73 (8.12)
	SpAM	22.23 (11.67)	0.00 (0.00)	0.047 (0.01)	1386.66 (243.80)
	Oracle	NA (NA)	NA (NA)	0.026 (0.00)	NA (NA)
Example 5	DDAC-SpAM	0.20 (1.15)	0.01 (0.10)	0.680 (0.39)	42.07 (2.51)
	DAC-SpAM	18.81 (13.69)	0.00 (0.00)	0.887 (0.16)	90.10 (6.64)
	SpAM	11.92 (9.00)	0.00 (0.00)	0.832 (0.15)	1514.38 (396.94)
	Oracle	NA (NA)	NA (NA)	0.643 (0.13)	NA (NA)

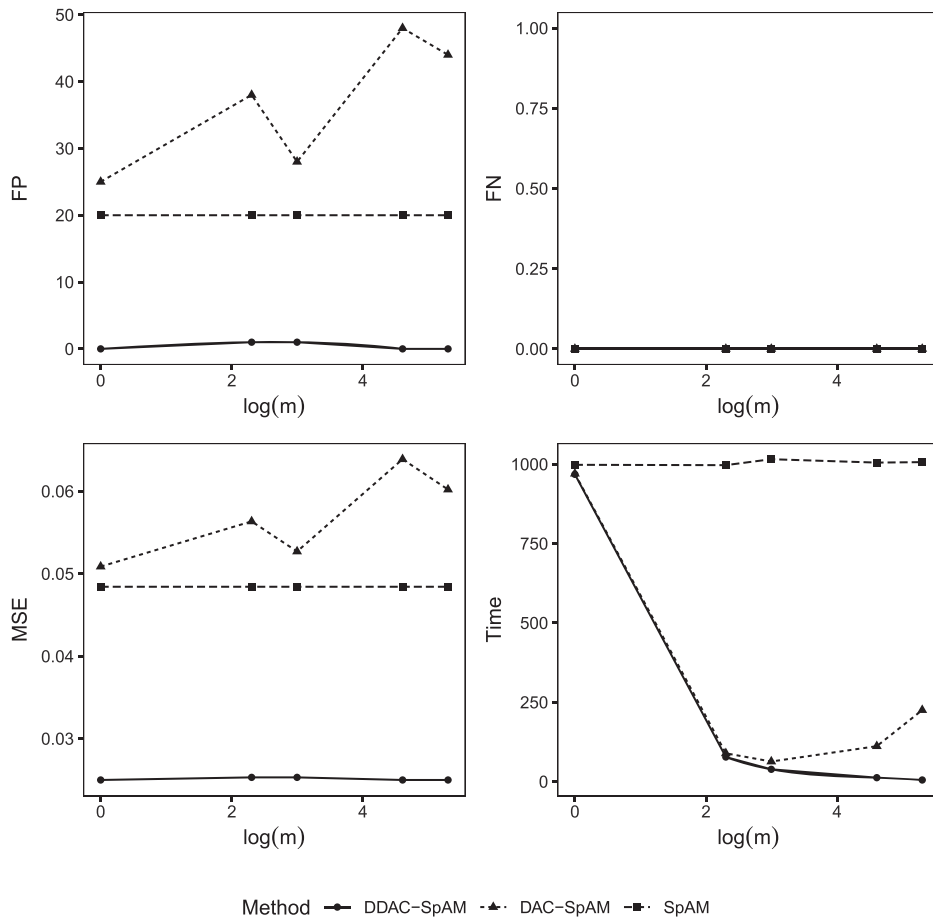


Figure 2. Performance of DDAC-SpAM, DAC-SpAM, and SpAM with different number of local machines.

additive model fitting. Overall, DDAC-SpAM excels at using all available computing resources, and is highly efficient and effective compared to existing algorithms for high-dimensional additive models.

### 6.3. Hypothesis Testing

In this section, we investigate the performance of the Chi-squared test in Algorithm 2 with simulated data, under a setting adapted from Example 3 in Section 6.1. The covariates  $X_j$  are generated in the same way:

$$X_j = \frac{W_j + tU_{\lfloor j/20 \rfloor}}{1 + t}, j = 1, \dots, p,$$

where  $W_1, \dots, W_p, U_1, \dots, U_{\lfloor p/20 \rfloor} \stackrel{iid}{\sim} \text{Uniform}(0, 1)$ . In the model, we include a parameter  $a$  to control the signal-to-noise ratio:

$$y_i = a [2.5g_1(x_{i1}) + 2.6g_5(x_{i2}) + g_6(x_{i3}, 2\pi) + g_7(x_{i4}, 2\pi)] + 0.5\epsilon_i.$$

As before, we fix the sample size  $n = 500$ , the dimension  $p = 10,000$ , the truncation size  $d_n = 5$ , and the number of machines  $m = 20$ . And we vary the parameter  $a$  from 0.1 to 1 in increments of 0.1. Finally, we set the significance level  $\alpha_0 = 0.05$ .

Note that due to the lack of formal hypothesis testing algorithms for high-dimensional sparse additive models, we focus

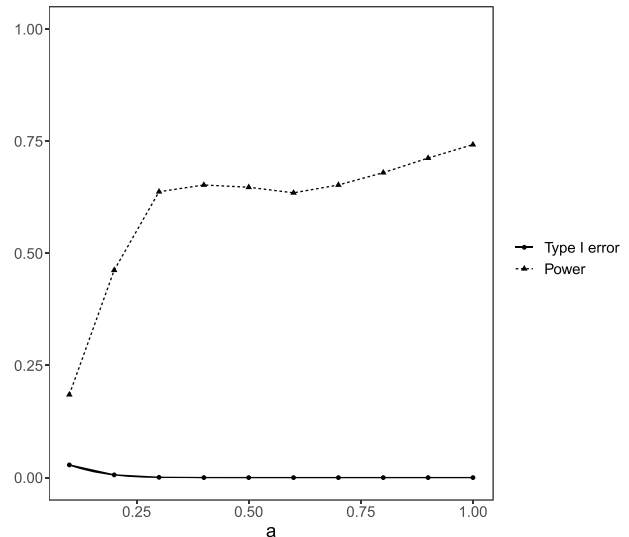


Figure 3. Type I error and power curves of the Chi-squared test in Algorithm 2.

on analyzing the performance of our method only. We report the averages of

$$\text{Type I error} = (p - 4)^{-1} \sum_{j=5}^p \mathbb{1}_{\{\mathcal{T}_j > F_5(0.95)\}}$$

and

$$\text{Power} = 4^{-1} \sum_{j=1}^4 \mathbb{1}_{\{\mathcal{T}_j > F_5(0.95)\}}$$

over 100 simulation runs in Figure 3. The Type I error curve stays constantly below the significance level of 0.05 across the range of  $a$ , whereas the power curve first shows a steep positive slope for small  $a$  values and then gradually increases as  $a$  continues to increase. The turning point occurs at  $a \approx 0.3$ , with  $SNR \approx 1.3$ .

## 7. An Application to Real Data

In this section, we compare the performances of DDAC-SpAM, SpAM, Deco-linear (Wang, Dunson, and Leng 2016), and lasso (Tibshirani 1996) on the meatspec dataset analyzed by Meier, Van de Geer, and Bühlmann (2009) and Yang and Zou (2017). The dataset was recorded by a Tecator near-infrared spectrometer which measured the spectrum of light transmitted through a sample of minced pork meat (Borggaard and Thodberg 1992; Thodberg 1993). It is available in the R package `faraway`. Our aim is to predict the fat content by absorbances which can be measured more easily. This original dataset contains  $n = 215$  observations with  $p = 100$  predictors which are highly correlated (Meier, Van de Geer, and Bühlmann 2009). In fact, the average correlation between different predictors is about 0.986. After all predictors are centered and scaled to have mean 0 and variance 1, we add 1900 simulated variables from joint distribution  $\mathcal{N}(0, \Sigma)$  as artificial noise terms, where  $\Sigma_{ii} = 1$  and  $\Sigma_{ij} = 0.98$  for  $i \neq j$ , for the purpose of mimicking the high-correlation among the “actual” predictors. Then, DDAC-SpAM, SpAM, Deco-linear, and lasso are applied to predict the fat content using these 2000 features. We use 10 machines for the DDAC-SpAM algorithm, where the features are distributed randomly. To compare the performances of all methods, we randomly split the dataset into a training set of 172 observations (80%) and a test set of 43 observations (20%), which also specifies the truncation size  $d_n = \lceil 0.1(172^{1/3} \log 172) \rceil = 3$ . We repeat the procedure 100 times. For each random split, we compute the number of predictors selected, the prediction errors on the test set, and the false positives among the 1900 simulated variables. Table 2 includes the average values and their associated robust standard deviations over 100 replications. To evaluate DDAC-SpAM’s dependency on the number of local machines, we conduct the same experiment with varying  $m$  values ( $m = 1, 5, 10, 20, 50$ ), and summarize the results in Table 3.

From Table 2, DDAC-SpAM outperforms Deco-Linear and lasso in terms of prediction errors. The performance of SpAM is superior to DDAC-SpAM, possibly because it can take advantage of the perfect independence between the original predictors and the 1900 new variables, while the decorrelation step in DDAC-SpAM inadvertently introduces correlation. Note that DDAC-SpAM selects significantly fewer predictors than

**Table 2.** Average prediction error (PE), model size (MS) and false positive count (FP) over 100 repetitions and their robust standard deviations (in parentheses) for DDAC-SpAM, SpAM, Deco-Linear and lasso.

Method	PE	MS	FP
DDAC-SpAM	0.337 (0.087)	13.84 (4.17)	7.52 (4.15)
SpAM	0.290 (0.084)	21.30 (5.03)	9.36 (5.25)
Deco-Linear	0.453 (0.114)	58.09 (23.00)	50.15 (19.14)
lasso	0.399 (0.107)	82.62 (32.99)	21.81 (11.18)

**Table 3.** Average prediction error (PE), model size (MS) and false positive count (FP) over 100 repetitions and their robust standard deviations (in parentheses) for DDAC-SpAM using  $m$  local machines.

$m$	PE	MS	FP
1	0.488 (0.108)	28.88 (14.12)	24.38 (13.37)
5	0.368 (0.085)	11.64 (3.53)	6.40 (3.55)
10	0.337 (0.087)	13.84 (4.17)	7.52 (4.15)
20	0.304 (0.078)	16.01 (4.50)	7.79 (4.38)
50	0.292 (0.077)	15.95 (3.84)	6.93 (3.73)

competing methods. Considering the high correlation among predictors and to provide a more parsimonious list, DDAC-SpAM could be a very worthwhile method for distributed feature selection. In Table 3, we observe the prediction errors of DDAC-SpAM steadily decrease as  $m$  increases. This is because with more machines used for distributed computing, it is more likely for correlated important features to be separated, making the consistent selection easier.

## 8. Discussion

In this article, we have developed a new feature-distributed learning framework named DDAC-SpAM for the high dimensional additive model. DDAC-SpAM makes predictors less correlated and more suitable for the further sparsistent variable selection. The experiments illustrate that this method not only reduces the computational cost substantially, but also outperforms the existing approach SpAM when covariates are highly correlated. This is the first work to combine the divide and conquer method with the high dimensional nonparametric model for feature-distributed learning. The results demonstrate that DDAC-SpAM is appealing through the lens of theoretical analysis, empirical performance and its straightforward implementation.

Given that we specifically approximate the additive components by truncated B-spline bases and then impose the sparsity penalty only, DDAC-SpAM framework is readily available for other smoothing method with the additive models, for example, smoothing splines (Speckman 1985) and sparsity-smoothness penalized approaches (Meier, Van de Geer, and Bühlmann 2009). Besides, extension to the generalized additive model can be an interesting topic for future research. Lastly, although DDAC-SpAM is currently designed to solve large- $p$ -small- $n$  problems, it can be naturally combined with a sample space partition step to deal with large- $p$ -large- $n$  problems. The details can be explored in future work.

## Supplementary Materials

The supplementary material consists of Lemma S.1–S.6 and the proofs of all lemmas, theorems, and corollaries.

## Acknowledgments

We thank the editor, the AE, and anonymous reviewers for their insightful comments which have greatly improved the scope and quality of the article.

## Disclosure Statement

The authors report there are no competing interests to declare.

## Funding

Zhou was supported by the State Key Program of National Natural Science Foundation of China (71931004) and National Natural Science Foundation of China (92046005) and the National Key R&D Program of China (2021YFA1000100, 2021YFA1000101). Feng was supported by NIH grant 1R21AG074205-01, NYU University Research Challenge Fund, a grant from NYU School of Global Public Health, and through the NYU IT High Performance Computing resources, services, and staff expertise.

## ORCID

Ruiyang Wu  <https://orcid.org/0000-0001-7417-9507>

Yang Feng  <https://orcid.org/0000-0001-7746-7598>

## References

- Aerts, M., Claeskens, G., and Wand, M. P. (2002), “Some Theory for Penalized Spline Generalized Additive Models,” *Journal of Statistical Planning and Inference*, 103, 455–470. [1933]
- Battey, H., Fan, J., Liu, H., Lu, J., and Zhu, Z. (2018), “Distributed Estimation and Inference with Statistical Guarantees,” *Annals of Statistics*, 46, 1352–1382. [1933]
- Borggaard, C., and Thodberg, H. H. (1992), “Optimal Minimal Neural Interpretation of Spectra,” *Analytical Chemistry*, 64, 545–551. [1942]
- Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2011), “Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers,” *Foundations and Trends in Machine Learning*, 3, 1–122. [1933]
- Cai, T. T., Zhang, A. R., and Zhou, Y. (2022), “Sparse Group Lasso: Optimal Sample Complexity, Convergence Rate, and Statistical Inference,” *IEEE Transactions on Information Theory*, 68, 5975–6002. [1934,1937]
- Chen, X., and Xie, M.-g. (2014), “A Split-and-Conquer Approach for Analysis of Extraordinarily Large Data,” *Statistica Sinica*, 24, 1655–1684. [1933]
- Chikuse, Y. (2003), *Statistics on Special Manifolds* (Vol. 174), New York: Springer. [1938]
- Downs, T. D. (1972), “Orientation Statistics,” *Biometrika*, 59, 665–676. [1938]
- Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004), “Least Angle Regression,” *The Annals of Statistics*, 32, 407–499. [1935]
- Foygel, R., and Drton, M. (2010), “Exact Block-Wise Optimization in Group Lasso and Sparse Group Lasso for Linear Regression,” *Statistics*, 1050, 11. [1935]
- Friedman, J., Hastie, T., and Tibshirani, R. (2010), “Regularization Paths for Generalized Linear Models via Coordinate Descent,” *Journal of Statistical Software*, 33, 1–22. [1939]
- Haris, A., Simon, N., and Shojaie, A. (2022), “Generalized Sparse Additive Models,” *Journal of Machine Learning Research*, 23, 1–56. [1934]
- Hastie, T., and Tibshirani, R. (1990), *Generalized Additive Models*, Chapman & Hall/CRC Monographs on Statistics & Applied Probability, New York: Taylor & Francis. [1935]
- He, Q., Zhang, H. H., Avery, C. L., and Lin, D. (2016), “Sparse Meta-Analysis with High-Dimensional Data,” *Biostatistics*, 17, 205–220. [1933]
- Huang, J., Horowitz, J. L., and Wei, F. (2010), “Variable Selection in Nonparametric Additive Models,” *Annals of Statistics*, 38, 2282–2313. [1933,1938]
- Jia, J., and Rohe, K. (2015), “Preconditioning the Lasso for Sign Consistency,” *Electronic Journal of Statistics*, 9, 1150–1172. [1934,1938]
- Khatri, C. G., and Pillai, K. C. S. (1965), “Some Results on the Non-central Multivariate Beta Distribution and Moments of Traces of Two Matrices,” *Annals of Mathematical Statistics*, 36, 1511–1520. [1936,1937]
- Koltchinskii, V., and Yuan, M. (2010), “Sparsity in Multiple Kernel Learning,” *The Annals of Statistics*, 38, 3660–3695. [1933]
- Lee, J. D., Liu, Q., Sun, Y., and Taylor, J. E. (2017), “Communication-Efficient Sparse Regression,” *The Journal of Machine Learning Research*, 18, 115–144. [1933]
- Lou, Y., Bien, J., Caruana, R., and Gehrke, J. (2016), “Sparse Partially Linear Additive Models,” *Journal of Computational and Graphical Statistics*, 25, 1126–1140. [1934]
- Meier, L., Van De Geer, S., and Bühlmann, P. (2008), “The Group Lasso for Logistic Regression,” *Journal of the Royal Statistical Society, Series B*, 70, 53–71. [1935]
- Meier, L., Van de Geer, S., and Bühlmann, P. (2009), “High-Dimensional Additive Modeling,” *The Annals of Statistics*, 37, 3779–3821. [1933,1935,1939,1942]
- Petersen, A., and Witten, D. (2019), “Data-Adaptive Additive Modeling,” *Statistics in Medicine*, 38, 583–600. [1934]
- Petersen, A., Witten, D., and Simon, N. (2016), “Fused Lasso Additive Model,” *Journal of Computational and Graphical Statistics*, 25, 1005–1025. [1933]
- Raskutti, G., Wainwright, M. J., and Yu, B. (2012), “Minimax-Optimal Rates for Sparse Additive Models Over Kernel Classes via Convex Programming,” *Journal of Machine Learning Research*, 13, 389–427. [1933,1934]
- Ravikumar, P., Lafferty, J., Liu, H., and Wasserman, L. (2009), “Sparse Additive Models,” *Journal of the Royal Statistical Society, Series B*, 71, 1009–1030. [1933,1934,1936,1938]
- Sadhanala, V., and Tibshirani, R. J. (2019), “Additive Models with Trend Filtering,” *The Annals of Statistics*, 47, 3032–3068. [1933,1934]
- Shi, C., Lu, W., and Song, R. (2018), “A Massive Data Framework for m-estimators with Cubic-Rate,” *Journal of the American Statistical Association*, 113, 1698–1709. [1933]
- Simon, N., and Tibshirani, R. (2012), “Standardization and the Group Lasso Penalty,” *Statistica Sinica*, 22, 983–1001. [1935,1936]
- Song, Q., and Liang, F. (2015), “A Split-and-Merge Bayesian Variable Selection Approach for Ultrahigh Dimensional Regression,” *Journal of the Royal Statistical Society, Series B*, 77, 947–972. [1934]
- Speckman, P. (1985), “Spline Smoothing and Optimal Rates of Convergence in Nonparametric Regression Models,” *The Annals of Statistics*, 13, 970–983. [1942]
- Stone, C. J. (1985), “Additive Regression and other Nonparametric Models,” *The Annals of Statistics*, 13, 689–705. [1933]
- Tang, L., Zhou, L., and Song, P. X.-K. (2016), “Method of Divide-and-Combine in Regularised Generalised Linear Models for Big Data,” arXiv preprint arXiv:1611.06208. [1933]
- Thodberg, H. H. (1993), “Ace of Bayes: Application of Neural Networks with Pruning,” Technical Report, Citeseer. [1942]
- Tibshirani, R. (1996), “Regression Shrinkage and Selection via the Lasso,” *Journal of the Royal Statistical Society, Series B*, 58, 267–288. [1935,1942]
- Van de Geer, S., Bühlmann, P., Ritov, Y., and Dezeure, R. (2014), “On Asymptotically Optimal Confidence Regions and Tests for High-Dimensional Models,” 42, 1166–1202. [1934,1937]
- Wang, X., Dunson, D. B., and Leng, C. (2016), “Decorrelated Feature Space Partitioning for Distributed Sparse Regression,” in *Advances in Neural Information Processing Systems*, pp. 802–810. [1934,1942]
- Wood, S. N. (2011), “Fast Stable Restricted Maximum Likelihood and Marginal Likelihood Estimation of Semiparametric Generalized Linear Models,” *Journal of the Royal Statistical Society, Series B*, 73, 3–36. [1935]
- Yang, J., Mahoney, M. W., Saunders, M., and Sun, Y. (2016), “Feature-Distributed Sparse Regression: A Screen-and-Clean Approach,” in *Advances in Neural Information Processing Systems*, pp. 2712–2720. [1934]
- Yang, Y., and Zou, H. (2015), “A Fast Unified Algorithm for Solving Group-Lasso Penalize Learning Problems,” *Statistics and Computing*, 25, 1129–1141. [1935]
- (2017), *glasso: Group Lasso Penalized Learning Using a Unified BMD Algorithm*. R package version 1.4. Available at <https://CRAN.R-project.org/package=glasso> [1939,1942]
- Yuan, M., and Lin, Y. (2006), “Model Selection and Estimation in Regression with Grouped Variables,” *Journal of the Royal Statistical Society, Series B*, 68, 49–67. [1936]
- Yuan, M., and Zhou, D. (2016), “Minimax Optimal Rates of Estimation in High Dimensional Additive Models,” *The Annals of Statistics*, 44, 2564–2593. [1933,1934]
- Zeng, D., and Lin, D. (2015), “On Random-Effects Meta-Analysis,” *Biometrika*, 102, 281–294. [1933]



- Zhang, Y., Duchi, J., and Wainwright, M. (2013), “Divide and Conquer Kernel Ridge Regression,” in *Conference on Learning Theory*, pp. 592–617. [1933]
- (2015), “Divide and Conquer Kernel Ridge Regression: A Distributed Algorithm with Minimax Optimal Rates,” *The Journal of Machine Learning Research*, 16, 3299–3340. [1933]
- Zhang, Y., Wainwright, M. J., and Duchi, J. C. (2012), “Communication-Efficient Algorithms for Statistical Optimization,” in *Advances in Neural Information Processing Systems*, pp. 1502–1510. [1933]
- Zhao, P., and Yu, B. (2006), “On Model Selection Consistency of Lasso,” *Journal of Machine Learning Research*, 7, 2541–2563. [1938]
- Zhao, T., Cheng, G., and Liu, H. (2016), “A Partially Linear Framework for Massive Heterogeneous Data,” *Annals of Statistics*, 44, 1400–1437. [1933]
- Zhou, Y., Porwal, U., Zhang, C., Ngo, H. Q., Nguyen, X., Ré, C., and Govindaraju, V. (2014), “Parallel Feature Selection Inspired by Group Testing,” in *Advances in Neural Information Processing Systems*, pp. 3554–3562. [1934]