Big Data Model Building using Dimension Reduction and Sample Selection

Lih-Yuan Deng[*]
Department of Mathematical Sciences, University of Memphis
and
Ching-Chi Yang[*]
Department of Mathematical Sciences, University of Memphis
and
Dale Bowman[*]
Department of Mathematical Sciences, University of Memphis
and
Dennis K. J. Lin [†]
Department of Statistics, Purdue University
and
Henry Horng-Shing Lu [‡]
Institute of Statistics, National Yang Ming Chiao Tung University

September 9, 2023

[*]This research was sponsored by the National Science Foundation under the award The Learner Data Institute (award #1934745). The opinions, findings, and results are solely the authors and do not reflect those of the funding agencies.

[†]This research of Dennis K. J. Lin was partially supported by National Science Foundation via Grant DMS-18102925.

[‡]This research of HHS Lu was partially supported by the Ministry of Science and Technology, Taiwan, under the grant numbers of 110-2118-M-A49-002-MY3 and 111-2634-F-A49-014-.

Abstract

It is difficult to handle the extraordinary data volume generated in many fields with current computational resources and techniques. This is very challenging when applying conventional statistical methods to big data. A common approach is to partition full data into smaller subdata for purposes such as training, testing, and validation. The primary purpose of training data is to represent the full data. To achieve this goal, the selection of training subdata becomes pivotal in retaining essential characteristics of the full data. Recently, several procedures have been proposed to select "optimal design points" as training subdata under pre-specified models, such as linear regression and logistic regression. However, these subdata will not be "optimal" if the assumed model is not appropriate. Furthermore, such subdata cannot be useful to build alternative models because it is not an appropriate representative sample of the full data. In this paper, we propose a novel algorithm for better model building and prediction via a process of selecting a "good" training sample. The proposed subdata can retain most characteristics of the original big data. It is also more robust that one can fit various response model and select the optimal model.

Keywords: dimension reduction, GAM, IBOSS, space-filling design.

1 Introduction

It is common to characterize big data with its 5V's: (1) "volume" for huge quantity (large *n*) or large number of variables (large *p*), (2) "variety" for various type, nature, and format of data, (3) "velocity" for ultra-high speed of data generation or collection, (4) "veracity" for the trustworthiness and quality of big data, and (5) "value" for its insights, usefulness and impact. Computational resources and techniques today are not capable of keeping up with the extraordinary volume of data being generated in many fields. Due to the overwhelming growth of data collection and big datasets, it can be challenging to extract useful information from big data with current computational resources. Like most statistical procedures, it is common to choose subdata from the big data for model estimation or inference/prediction for big data. Choosing a "good" subdata set as the training sample to build some suitable models is a critical issue. There are two "goodness" measures for the selected subdata: (a) ability to estimate the model parameters more precisely under a prescribed model for the big data, and (b) ability to find a suitable model representing the big data. To achieve (a), we should choose the subdata to maximize the "design/model efficiency" under the prescribed model. To achieve (b), we should choose the subdata to be as "similar" to the whole big data as possible. These "goodness measures" will lead to different subdata selection procedures which we will elaborate next.

The purpose of the first subdata selection procedure is the maximization of model efficiency so that it could reduce the variation of the parameter estimates under a prescribed model for the big data. A linear model is perhaps the most popular model in statistical and machine learning, for its simple implementation and interpretation to study the relation among features of big data. Because of a large number of input variables or usually a huge number of observations, the whole data may be too big for computers with limited memory. Under the setting of linear regression for big-data, Wang et al. (2019) proposed subdata selection methods based on D-optimality which is called information-based optimal subdata selection (IBOSS). Recently, Wang et al. (2021) proposed subdata selection methods based on A-optimality, which is called orthogonal subsampling (OSS). As a result, IBOSS is likely to select "boundary points" (because it is only choosing extreme points in each column) while OSS is likely to select "corner points". Therefore, such a subdata selection approach may not be feasible for fitting other more complicated models as the response function.

There are different subdata selection techniques proposed under different candidate models, e.g., quantile regression (Ai et al., 2021) and logistic regression (Cheng et al., 2020). As expected, one optimal subdata for a given model is not expected to remain "optimal" under alternative models. From the statistical/machine learning perspective, we would like the subdata selection techniques to explore possible models which approximate the underlying relationship between the predictors and the response variables. With many feasible candidate models to be explored, it is unrealistic to pre-specify any fixed model in advance.

The purpose of the second subdata selection procedure is the ability of finding a suitable model for the big data. To achieve this, we need to select subdata similar to the full data so that one can build a suitable model for the big data. Mak and Joseph (2018) proposed a method that retains the "similarity" of the subdata and the full data via some complicated measurement requiring significant computational power. Later, Joseph and Mak (2021) proposed a more efficient method to select the subdata filling the sample space via a k-mean cluster method.

In general, a "good" training sample should be representative of the original data, and we can achieve this by choosing the design points that are evenly spaced over the whole design space. See, for example, Fang et al. (2000). If the underlying simple (and unrealistic) linear model is true, selecting uniformly designed subdata will not be as "efficient" when compared to subdata selected by IBOSS/OSS which is likely to choose extreme/bounday points. Consequently, these "training subdata" would not be similar to the original big data and they are unable to adequately consider some more suitable and complicated models. In contrast, our proposed uniform design approach has several advantages: (a) it can yield "training subdata" similar to the original big data, (b) it can entertain various alternative models, and (c) according to our empirical evaluations to be presented later, it only loses a little model-matrix efficiency even for the simple (and unrealistic) linear model.

The most popular dimension reduction method is Principal Components Analysis (PCA), which is used to choose a suitable dimension of features to explore the underlying structure (see Abdi and Williams, 2010, for more detail). We propose to use PCA on the data to perform a subsampling procedure on its first few, say k ($<< p$), principal components (PCs) instead of subsampling over a large data dimension (p) space. For example, we consider a space-filling design that seeks design points, as a training sample, that is uniformly scattered (evenly spaced) on the PC space and/or on the whole design space. Such a training sample can be useful to entertain some possible complex models on the relation between the response and the inputs.

The choice of a possible model to relate the response variable with the variables from the original data or with "features" obtained from the given dimension reduction procedure is an issue. A linear model is clearly the most popular because of its simplicity and because it is easy to interpret and implement. However, the assumed linear model is rarely appropriate or correctly specified especially when the number of input dimensions is huge.

For big data, it is impractical and unnecessary to use the whole data set to build/estimate the models under consideration. Besides, the model might not be as simple as linear regression (see e.g., Furrer et al. 2006; Gramacy and Apley 2015). For example, fitting GAM can be infeasible or computationally inefficient if the number of input variables is large. Conventional methods of variable selection to choose a smaller number of "important variables" for GAM are also computationally inefficient. In this case, we can consider PCA to reduce the number of PC dimensions as input "features" for GAM. In this paper, we propose to use a powerful GAM procedure to estimate the actual response model. Finally, we provide an extensive empirical study on the same data used by Wang et al. (2019) for comparisons of various sampling plans. Additionally, we then keep the input variables and change only the response variable using a complex and popular response function. We evaluate and compare the performance of the GAM procedure under various sampling plans.

The rest of the paper is organized as follows. Section 2 reviews all existing approaches for selecting subdata. Then, the new methodology is proposed. Section 3 evaluates the performance of the proposed methods through simulated studies and real-world data. In Section 4, we summarize the key steps to construct an effective model for prediction: (1) carefully select the training sample, (2) apply dimension reduction techniques (e.g. PCA) to reduce the number of features and avoid over-fitting, and (3) build up a powerful GAM with the help of dimension reduction and a good sampling scheme to select a training sample.

2 Training Sample Selection procedures

2.1 Data and dimension reduction model

Consider n data observations $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$ where $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})$ is a covariate vector for the ith observation and y_i is the response variable. It is common to represent the dataset in a matrix form :

$$\mathbf{X} = \begin{pmatrix} x_{11} & \ldots & x_{1p} \\ x_{21} & \ldots & x_{2p} \\ . & & \\ . & & \\ x_{n1} & \ldots & x_{np} \end{pmatrix} = [X_1, X_2, \cdots, X_p], \quad \mathbf{Y} = \begin{pmatrix} y_1 \\ y_2 \\ . \\ . \\ y_n \end{pmatrix} \tag{1}$$

where the covariate matrix can be viewed as n row vectors of dimension p $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$ or p column vectors of dimension n $(X_1, X_2, \cdots, X_p)$ and $\mathbf{Y}$ is the response column vector of dimension n, $(y_1, y_2, \ldots, y_n)'$.

Let $\bar{\mathbf{X}}$ be a $n \times p$ matrix with each row as (row) sample average of $\mathbf{X} = \mathbf{1}'\mathbf{X}/n$, where $\mathbf{1} = (1, 1, \cdots, 1)'$ is a n-dimensional column vector of 1. The sample variance-covariance matrix is

$$\mathbf{S} = \frac{1}{n-1}(\mathbf{X} - \bar{\mathbf{X}})'(\mathbf{X} - \bar{\mathbf{X}}), \quad \bar{\mathbf{X}} = \frac{1}{n}\mathbf{1}'\mathbf{X}. \tag{2}$$

We assume the response vector $\mathbf{Y}$ can be approximated by a function of p covariates in $\mathbf{X}$, where p is too large to efficiently build a good approximation function. Typically, $\mathbf{X}$ may have a high correlation among some of the columns or some columns may have a non-linear relationship. Therefore, one can first use a dimension reduction method to reduce the data dimension to $k << p$ dimensions,

$$\delta(\mathbf{X}) \equiv \mathbf{X}^* \equiv [X_1^*, X_2^*, \cdots, X_k^*], \tag{3}$$

and it can build the connection between $\mathbf{Y}$ and $\mathbf{X}^*$ as

$$\mathbf{Y} = f(\mathbf{X}^*) + \epsilon, \tag{4}$$

where ϵ is a random error term and $f(\cdot)$ is the response function.

It is straightforward to use divide-and-conquer to (randomly) partition the big data into several blocks of subsets of the data (Li et al., 2013). Let $\mathbf{X}_i$ be the i^{th} subset having n_i data points in p-dimensional space, $i = 1, 2, 3, \cdots$. For each block, $\mathbf{X}_i$, perform the following subsampling procedure (to be described in detail later). For the remainder of this paper, we will drop the block index i and treat the partition as the "whole" data and denoted as $\mathbf{X}$ and $\mathbf{Y}$.

2.2 General models for response variable

The following model relating $\mathbf{Y}$ and $\mathbf{X}$:

$$\mathbf{Y} = f(\delta(\mathbf{X})) + \epsilon, \tag{5}$$

where ϵ is a random error term and $f(\cdot)$ is the response function.

Both the dimension reduction function $\delta(\cdot)$ and the response function $f(\cdot)$ need to be appropriately chosen. To further simplify the discussion, we consider a special form of $\delta(\mathbf{X}) = \mathbf{XC}$ and $f(\cdot)$ is a first order linear function. When $\mathbf{C} = \mathbf{I}$, we have (with $X_0 = 1$),

$$\mathbf{Y} = \sum_{i=0}^{p} \beta_i X_i + \epsilon. \tag{6}$$

For a general $\mathbf{C}$ of dimension $p \times k$, usually the "loading matrix" corresponding to the PCA dimensions, we have

$$\mathbf{Y} = \sum_{i=0}^{k} \beta_i X_i^* + \epsilon. \tag{7}$$

In general we can consider generalized additive model (GAM), proposed by Hastie and Tibshirani (1986) and Hastie and Tibshirani (1990), which replaces X_i in (6) by $g_i(X_i)$, and X_i^* in (7) by $g_i(X_i^*)$, where $g_i(\dot{)}$ is a smooth function to be estimated. Specifically, we consider a simple form of GAM as extensions of (6) using the original data dimensions (X_i) and (7) using the PCA dimensions (X_i^*) as

$$\mathbf{Y} = \sum_{i=0}^{p} g_i(X_i) + \epsilon. \tag{8}$$

and

$$\mathbf{Y} = \sum_{i=0}^{k} g_i(X_i^*) + \epsilon. \tag{9}$$

Next, we consider various subdata procedures and then propose a general subsampling procedure.

2.3 IBOSS subdata

Under the assumed model of (6), Wang et al. (2019) proposed a partition-based selection algorithm (Martinez, 2004) motivated by the upper bound of the information matrix of the linear regression estimator.

IBOSS Algorithm: Let s be the total sample size needed. For $r = s/(2p)$ perform the following steps:

1. For $x_{i1}, 1 \leq i \leq n$, include r data points with the r smallest x_{i1} values and r data points with the r largest x_{i1} values.

2. For $j = 2, \dots p$, excluding data points previously selected, choose r data points with the smallest x_{ij} and r data points with the largest x_{ij} values.

Potential weaknesses of IBOSS subdata procedure

There are concerns when using the algorithm proposed by Wang et al. (2019):

1. To implement IBOSS procedure, we may need to sort/find extreme points separately for all p dimensions of size n. It is well-known that the most efficient sorting algorithm of size n has the computing complexity of $O(n \log(n))$, and the computing complexity of $O(n)$ is to sample extreme points of size n. See, for example, Hoare (1961). To sample points over p-dimensional space of size n each, the total complexity would be (approximately) of the order $O(np)$. For large dimension p or a large number of observations n, the process of IBOSS becomes time-consuming. The IBOSS algorithm selects points from each covariate sequentially excluding data points previously selected. When the dimension p is large, there will be a large amount of data points excluded, especially when most of the columns are (usually) highly correlated.

2. The IBOSS sampling procedure is highly sensitive to possible outliers, especially for the situation of big data. Extreme points may not automatically be indications of outliers unless

they greatly deviate from the rest of the data. The justification behind IBOSS procedure is to choose the design points to maximize the model-matrix efficiency. However, our empirical results (shown later) indicate that there is a very little gain over other sampling designs. For example, it would not lose much model-matrix efficiency (and is more robust) to trim a few extreme points in each and every dimension before the implementation of IBOSS sampling procedure.

3. Another problem for IBOSS sampling is when some of the variables are ordinal categorical variables of more than 2 levels. In this case, IBOSS sample would leave out all data points with middle-level values. This is clearly unacceptable. There are several sampling schemes that can easily avoid this problem. For example, we can apply uniform systematic sampling on each of the p dimensions. We will discuss in detail this type of uniform sampling later.

4. Since IBOSS would choose extreme points in each of the p-dimensions, it can create a large gap among the selected points. Consequently, these sample points can only entertain a simple linear model between the response variable $\mathbf{Y}$ and the input variables $\mathbf{X}$. In theory, the IBOSS-selected design points can be optimal, if one can assume (unrealistically) that the "best" statistical linear model is known. In practice, the actual form of the response function, $\mathbf{Y} = f(\mathbf{X})$, is rarely known and the popular linear model can often be inappropriate especially when p is very large. The large gap in the IBOSS sample would make it infeasible to explore other possible and more complex models with non-smooth response functions having many local maxima/minima. As to be demonstrated later, any model estimation with such an IBOSS sample would fail to capture the actual shape of such a response function.

5. Model-matrix efficiency should not be a major concern for exploring big data. While the IBOSS sample is optimal if the assumed linear model in (6) is true, its actual percentage gain over other sampling methods can be very small. Because of the size of the big data, one can always choose more points to increase the model-matrix efficiency. Of course, it is common to compare the model-matrix efficiency with the same number of data points. The main purpose of choosing sample points, called the training sample is to fit the model, its parameters, and then make good predictions on other new (unused) data, called the test sam-

ple. It is difficult to find an appropriate model which is both useful, simple to estimate, and interpretable. Even if such a model can be found, a good model estimator/predictor would rely heavily on the characteristics of the training sample. Specifically, we need to choose the training sample which is similar to the test sample and the whole big data. Clearly, the IBOSS subdata is in no way similar to the whole big data.

In summary, it is essential that the subsample (training sample) is chosen to be representative of the full data set so that additional model analysis can be considered. Next, we discuss a very general sampling procedure that should be able to choose a "better" training sample.

2.4 TSF-u(PCA): Trimmed Space-filling sampling on PCA

2.4.1 Trimmed-Space-Filling (TSF) design/sampling

A space-filling (SF) design seeks design points that are uniformly scattered (evenly spaced) on the domain. Such a design can be useful to detect possible model assumption violations on the assumed relationship between the response and the inputs. For example, suppose one assumes that the relationship between the response and a single input is essentially linear when, in fact, it is highly nonlinear. To detect such model violations, one may need to choose design points evenly over the whole design space. See, for example, Fang et al. (2000).

Let $D_z = \left\{z_1, z_2, \cdots, z_n\right\}$ be the set of n units and we would like to choose a sample of size s which are evenly distributed over the set D_z. For the i^{th} data point, let r_i be its rank (according to the increasing value of z_i) and d_i be some distance measure (e.g. Euclidean distance) to its central location (e.g. mean or median). For IBOSS scheme on the specified column in the whole data matrix, it would choose extreme data points, *equally* on both ends, with $r_i \leq s/2$ or $r_i > n - s/2$. A slight variation of IBOSS scheme is to choose data points that are far apart from its center with large values of d_i. Clearly, for symmetric data (on the z_i values) both schemes would yield similar data points. For heavily skewed distribution, the difference could be substantial. As previously discussed, such data points may be "optimal" under a certain restricted linear model on the response variable and input variables. On the other hand, it is more likely to contain outliers and it is less likely to build various alternative models.

To avoid the possible problems as discussed above, we propose to trim the data set by removing some small (user-controlled) number of extreme data from the original whole data and then apply some space-filling scheme to choose data points uniformly/evenly spaced/distributed over the remaining trimmed data set. There are several space-filling schemes that can be considered.

1. The simplest and most efficient method is using simple random sampling. While its sampling points are randomly distributed over the whole data set, it may not guarantee they are evenly spaced (especially when the sample size s is small) over the high-dimensional data space.

2. Another approach is using systematic sampling (see, e.g., Cochran 1977) where the set is divided evenly into s subsets of size $t = n/s$ in the increasing order of r_i. Choose a random starting entry, say, uniform random number w from $Unif(1, 2, \cdots, t)$. Choose the s points with indices, $w + (j - 1)t$, for $j = 1, 2, \cdots, s$ from s subsets. The advantage of this scheme is that sample units are indeed equally spaced (of length t) in the values of i^{th} rank, r_i. However, it has a limited number of possible samples that can be chosen.

3. Finally, we can modify the above systematic sampling scheme with more flexibility to choose the random sample. The main idea is to use random index (w_j from $Unif(1, 2, \cdots, t)$) to choose independently in each of the s subsets. Specifically, we choose the s points with indices, $w_j + (j - 1)t$, for $j = 1, 2, \cdots, s$ from s subsets. While sample units may not be equally spaced in the values of the i^{th} rank, r_i, the maximum space is at most $2 \times t$ and exactly one unit is chosen within each of the s subsets. This sampling scheme is like a well-known stratified random sampling scheme by dividing the data population into s strata, each of size t, according to the values of r_i. Choose one unit each from the s subsets. Compared with simple random sampling, the current method should yield a more evenly spaced sample in the values of the i^{th} rank, r_i. Hence, we will refer to this scheme as the space-filling (SF) scheme in this paper.

2.4.2 TSF-u(PCA) sampling procedure

Our proposed method, TSF-u(PCA), is to apply the proposed space-filling scheme on the principal components to select the sub-sample as briefly described next.

Let D be the set of indices left for a possible future selection and S be the set of points being selected. Initially, D is the set of indices corresponding to whole data, and S is an empty set. Both D and S will be continuously updated during the data selection process.

Let s_i denote the sample size for the i^{th} PC and it is reasonable to choose s_i, to be decreasing to account because of the decreasing variation in the i^{th} PC. However, it is simpler to set the sample size for each PC the same, $s_i = s/k$, where s is the total sample needed and k is the effective PCA dimensions. For most cases, k can be expected to be much smaller than p. In fact, for the purpose of efficient subdata selection, it is unnecessary to choose a very large k simply to account for a very high percentage of “total variation” of the data matrix. In practice, we recommend $k \leq 10$ for the subdata selection as described below.

1. Repeat this process from $i = 1$ to $i = k$, that is, the first k principal components with the set S representing the final data that is selected.

2. Find the principal component (PC_i) of $\mathbf{X}$ which is a linear combination of the p columns in $\mathbf{X}$ with the largest variation.

3. To avoid possible outliers, we update D and exclude the data points corresponding to a few extreme values of PC_i.

4. Perform a uniform design/space-filling procedure to select s_i data points based on the values of PC_i from the possible data set indexed by S. Update the set S and D with those data selected.

It is interesting to note that the above TSF-u(PCA) can be easily extended to TSF-u($\delta(\mathbf{X})$), where $\delta(\mathbf{X})$ in (3) is any general dimension reduction procedure including even its original data with $\delta(\mathbf{X}) = \mathbf{X}$.

2.4.3 Comparing IBOSS and TSF-u(PCA) sampling scheme

The proposed TSF-u(PCA) sampling scheme can resolve some of the problems discussed earlier for IBOSS subdata procedure:

1. TSF-u(PCA) is more robust than IBOSS (and simple random sampling), because it is trimming possible outliers with extreme values excluded in each of the k PC dimensions. The amount of "trimming" is user-controlled.

2. TSF-u(PCA) sampling scheme is based on k PCA-dimensions with many possible sample selections of the data points. IBOSS subdata is based on p dimensions of the original dataset $\mathbf{X}$. In addition to the fact that $k << p$, IBOSS is less efficient because it is likely to remove lots of the sampled points for correlated data. Unless we change the selection order of the data columns, IBOSS sample is deterministic with fixed data and s (sample size).

3. Like the simple random sampling scheme, the TSF-u(PCA) sample scheme is expected to choose data points that are evenly distributed in both the space of the original data dimension and the PCA dimension. In contrast, IBOSS will choose extreme points and it will create "huge holes" among the selected data. The IBOSS subdata is highly "dis-similar" to the original whole population and it is unlikely to be a good representative sample when considering other complicated models.

2.4.4 TSF-v(PCA): improved vector version of TSF-u(PCA)

The key characteristic of TSF-u(PCA) sampling procedure is the use of a Trimmed Space Filling procedure (TSF) on the matrix of size $n \times k$ corresponding to its effective k PCA dimensions. While IBOSS needed p iterations, TSF-u(PCA) needs $k(< p)$ iterations. Like the IBOSS scheme, TSF-u(PCA) sampling scheme is choosing a data point based on the values of its specific dimension individually, not jointly. Next, we discuss an improvement on the TSF-u(PCA) sampling scheme which requires only one iteration by considering the data vector jointly.

For simplicity, we discuss first the improved procedure using the original data matrix $\mathbf{X}$ as in (1), denoted as TSF-v($\mathbf{X}$). The same process can also be applied for PCA dimension which is denoted as TSF-v(PCA). For the i^{th} data point, $\mathbf{x}_i$, in the data matrix $\mathbf{X}$, we define the popular distance measure between $\mathbf{x}_i$ and its mean vector, $\bar{\mathbf{X}}$, as

$$D_i^2 = (\mathbf{x}_i - \bar{\mathbf{X}})'\mathbf{S}^{-1}(\mathbf{x}_i - \bar{\mathbf{X}}), \tag{10}$$

where $\mathbf{S}$ is the variance-covariance matrix defined in (2). D_i^2 is the Mahalanobis distance in the area of multivariate analysis as proposed in Mahalanobis (1936). See, for example, Anderson (2003).

For big data with a large number of dimensions (p), it is time-consuming to compute $\mathbf{S}^{-1}$ and we recommend replacing $\mathbf{S}$ with its diagonal matrix $\mathbf{D} = Diag(\mathbf{S})$, which makes it much easier to compute its inverse as

$$D_i^2 = (\mathbf{x}_i - \bar{\mathbf{X}})'\mathbf{D}^{-1}(\mathbf{x}_i - \bar{\mathbf{X}}), \quad \mathbf{D} = Diag(\mathbf{S}) \tag{11}$$

Finally, we propose the following new scheme, called TSF-v(X):

1. Compute and sort (i, D_i^2), $i = 1, 2, \cdots, n$ in increasing order of D_i^2.

2. Trim fixed (user-controlled) number/percentage of pairs, (i, D_i^2), with large value D_i^2.

3. Using D_i^2, apply the uniform-space filling on the remaining dataset to find s data points.

When the proposed scheme is applied to the case with $\mathbf{X}$ being the PCA columns, called TSF-v(PCA). In this case, the PCA columns are orthogonal and its variance-covariance matrix $\mathbf{S}$ would be a diagonal matrix $\mathbf{D}$ in Equation (10) and (11). Clearly, TSF-v(PCA) is an improvement over TSF-u(PCA) where k iterations are required and it needs to deal with the steps of removing sample duplications. Additionally, TSF-v(PCA) has more intuitive appeal because a k-dimension vector of all k PCA components is used jointly in the sampling scheme. It would be interesting to study any additional advantages, in theory, and/or in practice. Both TSF-u(PCA) and TSF-v(PCA) share the same characteristics that both will remove some possible outliers and both will choose the data points evenly over the space in the PCA dimensions or the original data dimensions.

While the Mahalanobis distance was used for the purpose of subsampling, it is just one of the distance measures that can be used. For example, Liu et al. (1999) discussed some measures of "data-depth", including Mahalanobis distance, for the multivariate data. Clearly, we can consider a similar sub-sampling procedure based on those data-depth measures on both the original data space or PCA space.

To simplify the comparison with IBOSS scheme, we use the initially proposed scheme, TSF-u(PCA), for the remaining of this paper.

3 Empirical Evaluation and Simulation

In this section, we first perform an extensive simulation to evaluate the computing efficiency between the proposed method and IBOSS with various combinations of n, p, k to select subdata of size s. Next, we evaluate and compare the statistical properties of choosing a training sample via IBOSS and the proposed method on a real data set discussed in Wang et al. (2019). We also demonstrate the importance of selecting a good training sample on the model-building performances in the original data set as well as a simulated response variable replacing the original response variable.

3.1 Performance and computational time comparison

To assess the computational efficiency of obtaining subdata using IBOSS and TSF-u(PCA), we perform a large-scale simulation study with combinations of key factors that affect the computational time of IBOSS and TSF-u(PCA): (i) the data size n and (ii) the number of input variables p. The simulations were conducted using the High-Performance Computing facility at the University of Memphis with PowerEdge C6420 Compute Nodes, featuring Intel Skylake Gold 6148 Processors, 192 GB DDR4 RAM, and EDR Infiniband. Specifically, we consider combinations of $n = 10^5$, $n = 10^6$, $n = 10^7$ and $p = 125$, $p = 250$, $p = 500$. A data frame of size $n \times p$ will be generated from the uniform distribution $U(0, 1)$ independently. Another key factor is the effective PC dimensions k. As explained previously, there is no need to set a large k simply to account for a high percentage of "total variation" of the data matrix. In this study, we choose $k = 2$, $k = 4$, and $k = 6$. For the subdata size s, it is set as 1000, to satisfy IBOSS' requirement that the subdata size must be at least twice the number of input variables (i.e., $2p$). To account for the potential variability, we repeat each simulation 100 times and calculate the average computational time.

Finally, we adopt the implementation of IBOSS using the R package developed by the original authors, which is available at `https://github.com/Ossifragus/IBOSS.git`.

Under various combinations as described, the average computational time is shown in Figure 1. A few conclusions can be made: (1) when the sample n is not very large (e.g. $n = 10^5$ or $n = 10^6$), there is little or no difference in computing time for two methods under various choices of p and k, and (2) when the sample n is large (e.g. $n = 10^7$), the IBOSS subdata selection is less efficient especially when p is also large. This is expected because IBOSS procedure needs to sort p columns of length n.

Next, we compare the design efficiency of the selected subdata between IBOSS and TSF-u(PCA) across various scenarios. IBOSS is optimal under the assumption, although unrealistic, that all p input variables are active in the first-order linear model. However, practical situations often involve inactive input variables and/or require more suitable complex models beyond the first-order model. In such scenarios, IBOSS might not exhibit a substantial performance advantage over our proposed method, as illustrated in the upcoming simulation study. In particular, we choose population size $n = 10^5$, $p = 500$, the subdata size $s = 1000$, and only $t = 3$ are important variables. We repeat the process 100 times and compute the design efficiency measures, $|X'X|$, under reduced model of $t = 3$ active variables with both (i) the first-order linear model and (ii) second-order polynomial model. The results are shown in Figure 2. As shown in Figure 2(a), under (i), differences in design efficiency measure between the IBOSS subdata and the TSF-u(PCA) subdata are not significant. Specifically, the BOSS subdata under (i) outperformed the TSF-u(PCA) subdata in only 56 instances out of the 100 repetitions. Moreover, under (ii), the difference is insignificant, as shown in Figure 2(b). In summary, TSF-u(PCA) demonstrates greater computational efficiency compared to IBOSS, while maintaining a design efficiency equivalent to that of IBOSS.

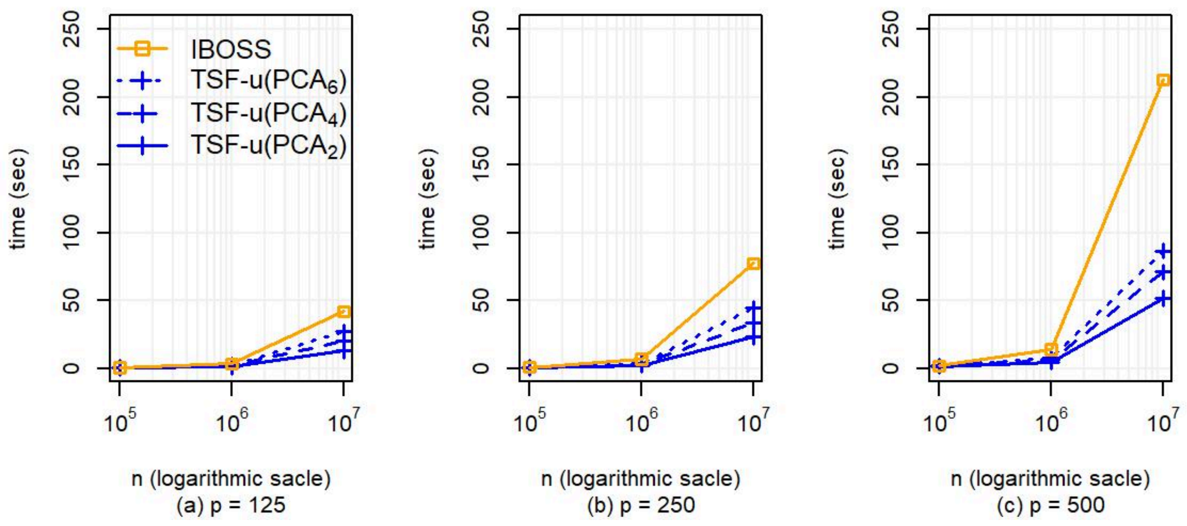
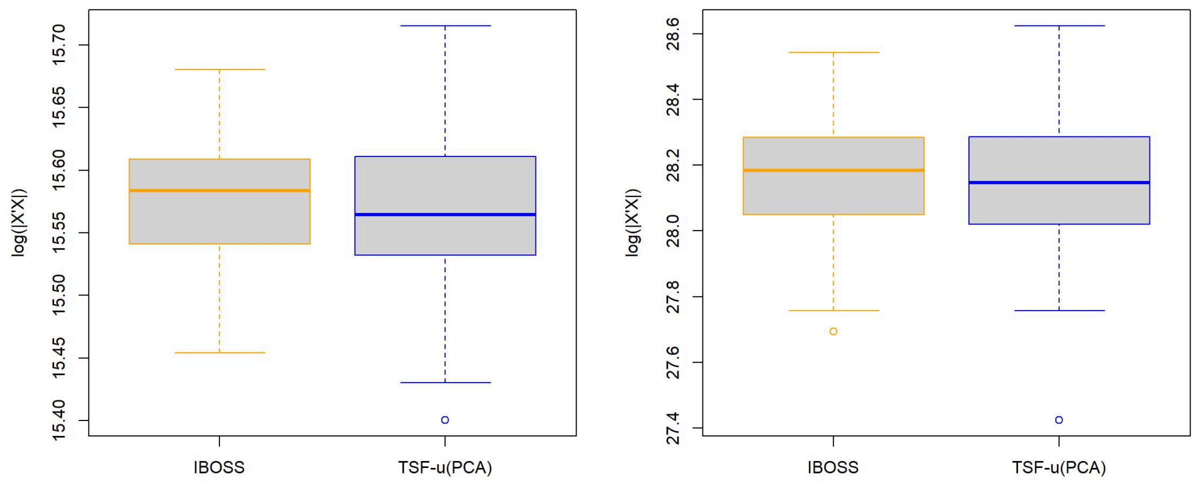


Figure 1: Computational time comparisons with different methods, *n*, and *p*.



(a) First-order model.

(b) Quadratic model.

Figure 2: Boxplot of the logarithm of the esign efficiency obtained from 100 subdata sets using IBOSS and TSF-u(PCA).

Next, we consider other more important criteria (e.g. design efficiency or model building ability from subdata selected) with some real data.

3.2 EDA on Chemical Sensors Data

For the comparison, we use the same example considered in Wang et al. (2019) where chemical sensors data were collected to develop and test strategies to solve a wide variety of tasks. This is in favor of their setup. Specifically, Fonollosa et al. (2015) discussed using the data to develop algorithms for continuously monitoring or improving the response time of sensory systems. The data were collected at the ChemoSignals Laboratory in the BioCircuits Institute, University of California San Diego. It contains the readings of 16 chemical sensors exposed to the mixture of Ethylene and CO at varying concentrations in air. Each measurement was constructed by the continuous acquisition of the sixteen-sensor array signals for a duration of about 12 hours without interruption.

To make a direct comparison with Wang et al. (2019), we followed their data processing steps without additional citations/explanations. They used the reading from the last sensor as the response (Y) and readings from other sensors as covariates. Since trace concentrations often have a log-normal distribution, a log-transformation of the sensors readings is used. Readings from the second sensor are not used in the analysis because about 20% of the values are negative for some reasons unknown. In total, there are $p = 14$ covariates, denoted as $Z_1, Z_2, \cdots, Z_{14}$ and they excluded the first $20,000$ data points corresponding to less than 4 min of system run-in time from the full data used containing $n = 4,188,261$ data points.

3.2.1 Preliminary data analysis

Due to the problem of over-plotting, the full big-data population of size respectively $4,188,261$ data points is too large for meaningful data exploration, method comparison, and data visualization. In practice, it is unnecessary to use the original whole big-data set and we can obtain a reasonable smaller size "sub-population" to represent the whole data. Following Wang et al. (2019), we create the "sub-population" consist of the response variable (Y) and 14 covariates ($Z_i, i = 1, 2, \cdots, 14$) from a simple random sample of size $n = 10,000$ from the population of size $4,188,261$ data points. This is consistent with the common practice for handling big-data where we can apply the divide-and-conquer method to partition the original big-data into several blocks of manageable size n.

As a simple exploration, we divide the 14 predictors into two groups according to their correlation coefficients. The first group of covariates of high correlation (≥ 0.98) is

$$G_1 = \left\{ Z_2, Z_3, Z_6, Z_7, Z_{10}, Z_{11}, Z_{14}, \right\} \tag{12}$$

where any columns in G_1 are highly correlated with Y, and any two of the columns in G_1 are also highly linearly correlated. The remaining columns form the second group of covariates as

$$G_2 = \left\{ Z_1, (Z_4, Z_5, Z_{12}, Z_{13}), (Z_8, Z_9) \right\}, \tag{13}$$

where any two of the columns in G_2 are less linearly correlated except those grouped as (Z_4, Z_5, Z_{12}, Z_{13}), (Z_8, Z_9). The scatter plots between Z variable in first group G_1, G_2 with Y are given in Figure 3 and 4, and the linear pattern between variables can be shown within the groups.

Wang et al. (2019) claimed that a linear model seems appropriate for the log-transformed readings between the response (Y) and covariates $Z_i, (i = 1, 2, \cdots, 14)$. It should be noted that from the plot between Y and $Z's$ in group G_1, it is reasonable to consider a much simple linear model between Y with just one of the Z_i in G_1. Later, we will re-evaluate the whole procedure by simulating a new Y with a complicated response function from some subset of the Z_i and no linear models would be suitable.

3.2.2 Empirical comparison between IBOSS and TSF-u(PCA) method

Following Wang et al. (2019), we choose a training sample using the IBOSS scheme of size $s = 20 \times p = 280$. To implement the sampling scheme, TSF-u(PCA), we perform PCA on the data and we find that its effective dimension is 4, a significant reduction from its original dimension of 14. The scatter plots of the selected data points projected on selected pairs of Z_i dimensions are shown in Figures 5 and 6. In particular, we use the following color/symbol schemes for different data points:

1. light gray color for the original data points,

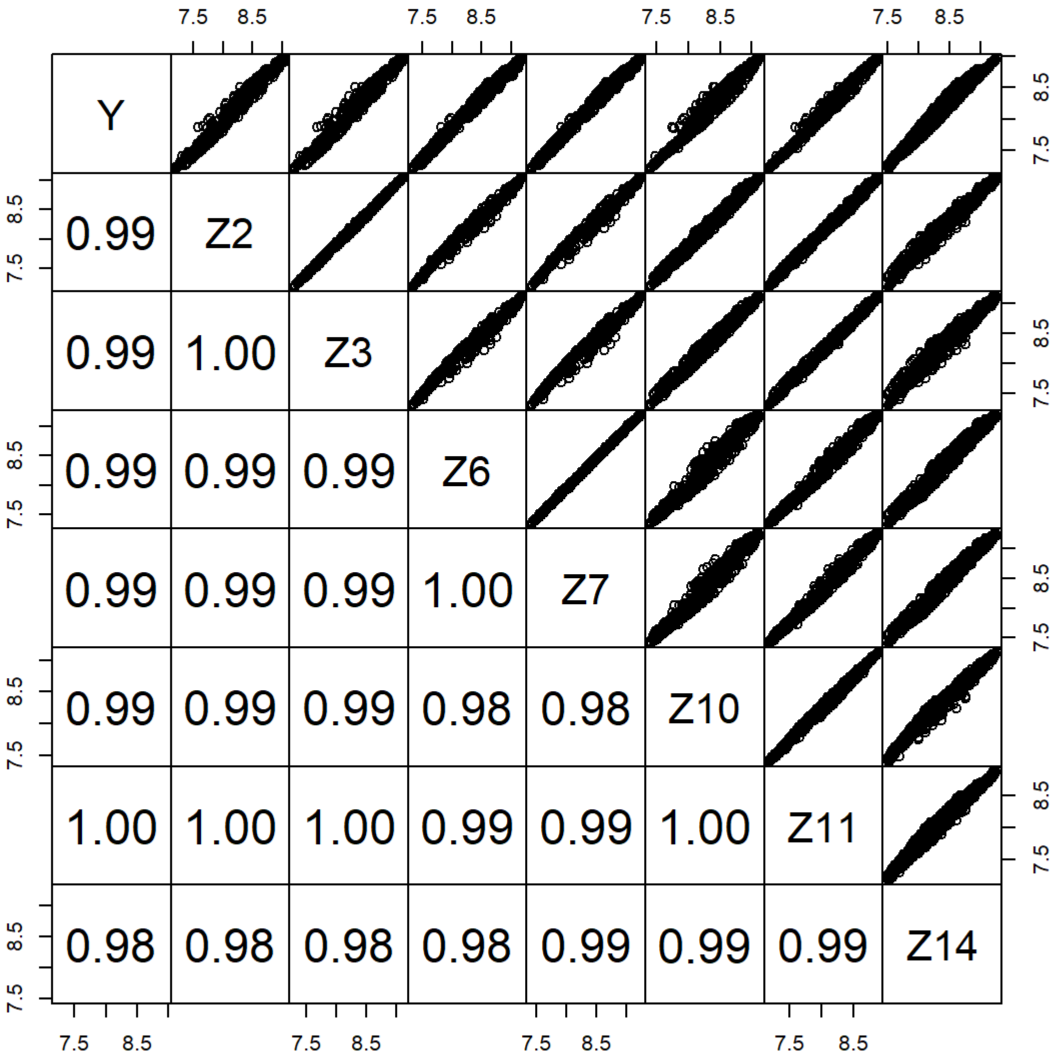


Figure 3: Pairwise plots of Y and Z's in group G_1 with strong linear correlation

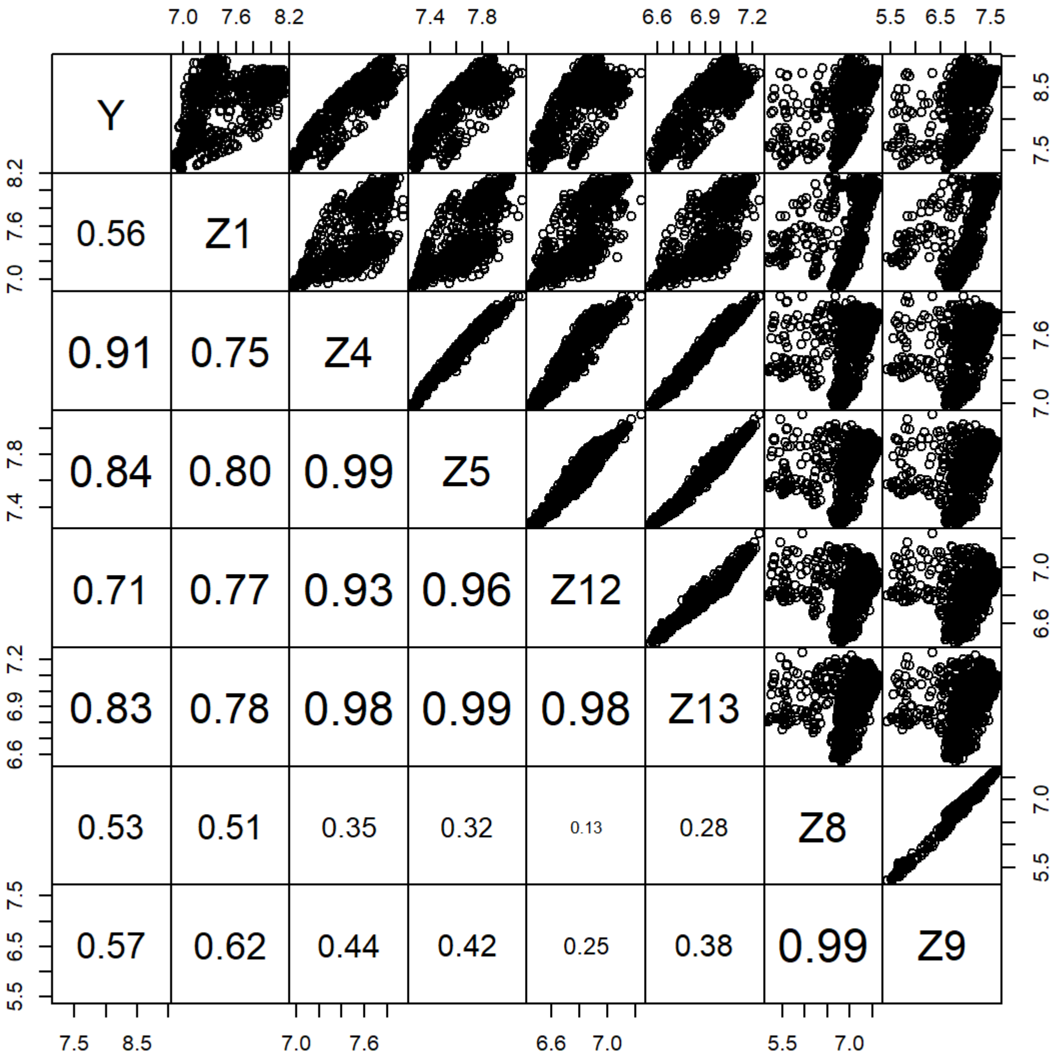


Figure 4: Pairwise plots of Y and Z's in other group G_2

2. red color circle for the subdata points from IBOSS,
3. blue color triangle for the sampled points from TSF-u(PCA).

As expected, unlike the sampled points from TSF-u(PCA), the subdata points from IBOSS failed to cover the original data space.

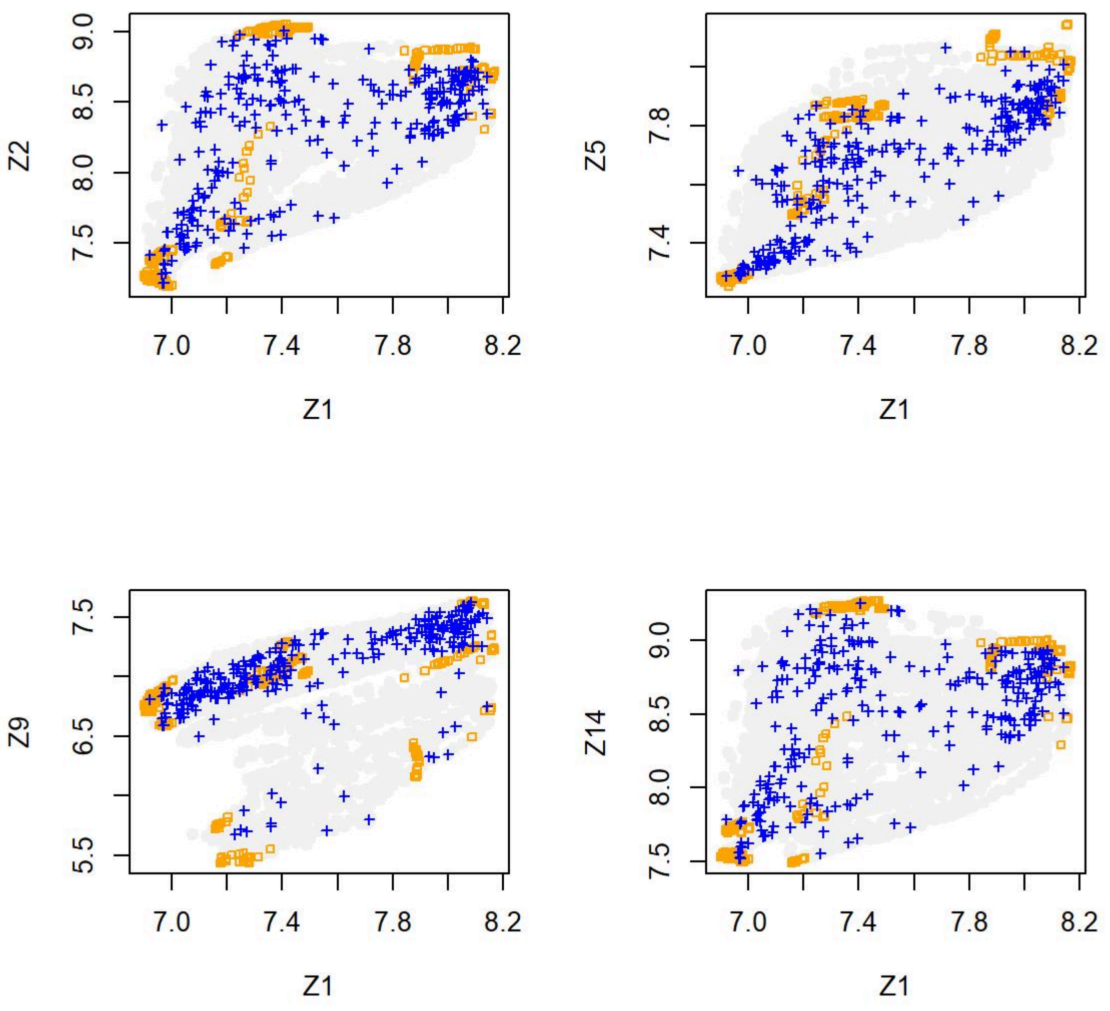


Figure 5: Scatter plot of Z_1 vs Z_2, Z_5, Z_9, Z_{14}, original data (light gray), subdata selected by IBOSS (marked as □) and sample by TSF-u(PCA) (marked as $+$).

We then implement our sampling scheme, TSF-u(PCA) (uniform sampling based on 4 PCA

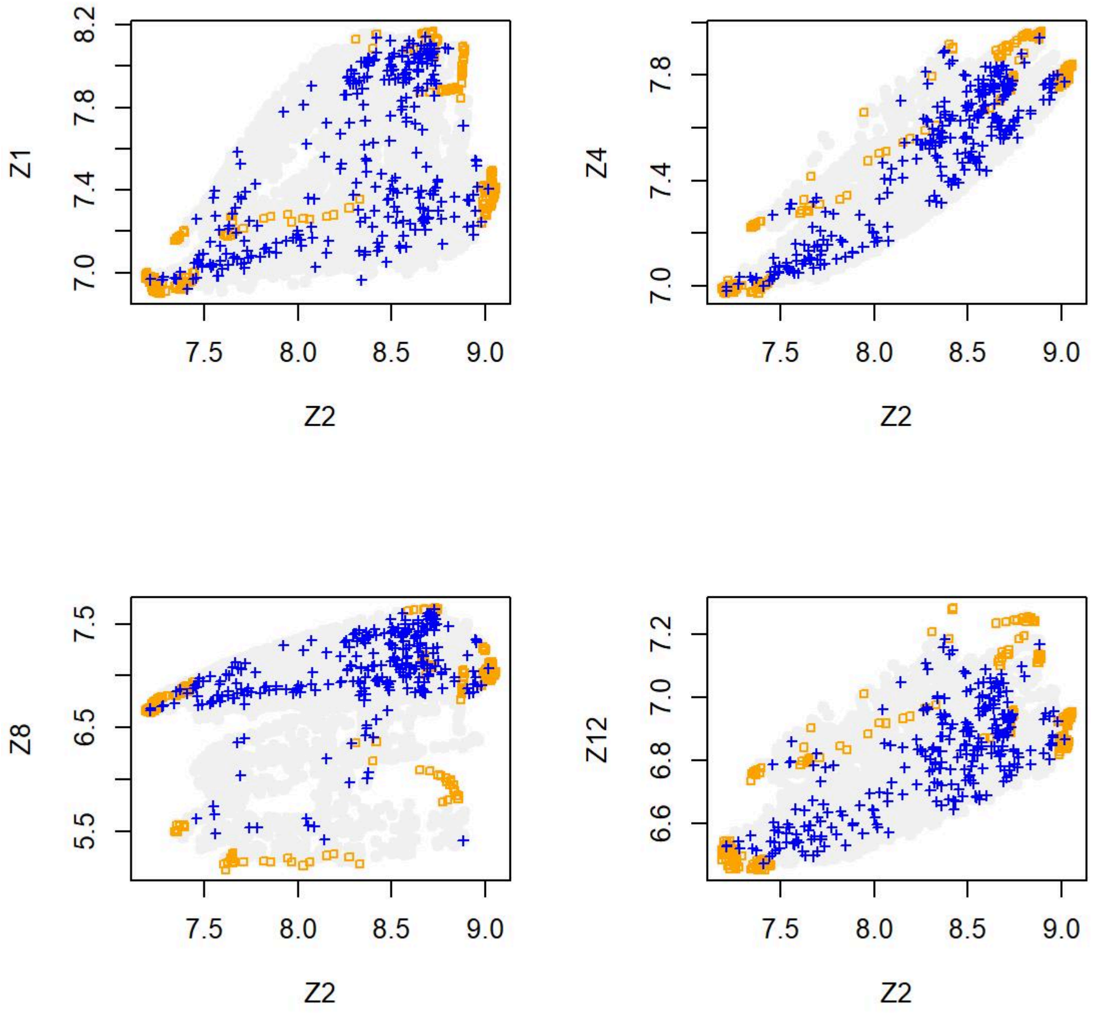


Figure 6: Scatter plot of Z_2 vs Z_1, Z_4, Z_8, Z_{12} original data (light gray), subdata selected by IBOSS (marked as $\square$) and sample by TSF-u(PCA) (marked as $+$).

columns), we choose 280 (random) points from the same population. We evaluate the performance of the D-OPT IBOSS subdata, denoted as $\mathbf{X}_b$, and sampled by TSF-u(PCA) with the same number of points, denoted as $\mathbf{X}_u$. As noted previously, IBOSS subdata will yield a fixed subdata $\mathbf{X}_b$ and we can compute its D-efficiency $\text{deff}_b = \log\det(\mathbf{X}_b'\mathbf{X}_b)$. On the other hand, there are many ways to choose subsample $\mathbf{X}_u$ and compute the corresponding $\text{deff}_u = \log\det(\mathbf{X}_u'\mathbf{X}_u)$. For a fair comparison, we select 500 random sample $\mathbf{X}_u$ and find the "distribution" of deff_u. For IBOSS subdata, $\text{deff}_b = -6.767$ which is smaller than the sample average of 500 values of $\text{deff}_u = -6.519$. According to our simulation evaluation, the percentage of $\text{deff}_u > \text{deff}_b$ is 0.726. If we choose another measure on the corresponding variance-covariance matrix in (2), $\text{deff}_b = \log\det(\mathbf{S}_b)$ and $\text{deff}_u = \log\det(\mathbf{S}_u)$. Among the 500 iterations, we find 0.996 cases that $\text{deff}_u > \text{deff}_b = -97.71$ with the average of 500 values of deff_u is -96.43. It is interesting to note that D-OPT IBOSS subdata yielding the subdata $\mathbf{X}_b$ does not unnecessarily outperform over samples by TSF-u(PCA).

Next, we compare the variance-covariance matrices, in (2), by subdate by IBOSS, subsample from TSF-u(PCA) and the original data $\mathbf{X}$ for variables in G_1 and G_2 in (12) and (13). The variance-covariance matrix of IBOSS subdata $\mathbf{S}_b$, original data $\mathbf{S}_o$, TSF-u(PCA) subsample $\mathbf{S}_u$ are shown in Table 1.

In table 1, IBOSS subdata clearly has a different structure in terms of the corresponding variance-covariance matrices. Compared with the IBOSS subdata, TSF-u(PCA) subsample clearly has a similar structure with that of the original data in terms of the corresponding variance-covariance matrices.

3.3 Simulated response of Chemical Sensors Data

In the previous example, the reading from the last sensor was chosen as the response (Y) and readings from other sensors as covariates. Consequently, Y is highly correlated with several input variables and its model/analysis can be too simplistic. It would be interesting to simulate a new response variable, Y, while keeping the original input variables. Therefore, the PCA dimensions, its TSF-u(PCA) sampling procedure as well as the IBOSS subdata are unchanged.

IBOSS subdata, Group G_1								IBOSS subdata, Group G_2							
X	Z2	Z3	Z6	Z7	Z10	Z11	Z14	X	Z1	Z4	Z5	Z8	Z9	Z12	Z13
Z2	0.62	0.61	0.63	0.62	0.57	0.57	0.58	Z1	0.184	0.147	0.118	0.057	0.088	0.097	0.099
Z3	0.61	0.60	0.62	0.61	0.56	0.56	0.57	Z4	0.147	0.162	0.123	0.048	0.075	0.096	0.103
Z6	0.63	0.62	0.64	0.63	0.58	0.58	0.59	Z5	0.118	0.123	0.096	0.023	0.047	0.077	0.080
Z7	0.62	0.61	0.63	0.63	0.57	0.57	0.58	Z8	0.057	0.048	0.023	0.371	0.314	-0.009	0.017
Z10	0.57	0.56	0.58	0.57	0.53	0.53	0.54	Z9	0.088	0.075	0.047	0.314	0.277	0.016	0.038
Z11	0.57	0.56	0.58	0.57	0.53	0.53	0.54	Z12	0.097	0.096	0.077	-0.009	0.016	0.064	0.064
Z14	0.58	0.57	0.59	0.58	0.54	0.54	0.55	Z13	0.099	0.103	0.080	0.017	0.038	0.064	0.067
original data, Group G_1								original data, Group G_2							
X	Z2	Z3	Z6	Z7	Z10	Z11	Z14	X	Z1	Z4	Z5	Z8	Z9	Z12	Z13
Z2	0.18	0.18	0.19	0.19	0.17	0.17	0.17	Z1	0.152	0.073	0.064	0.087	0.093	0.049	0.049
Z3	0.18	0.17	0.19	0.19	0.16	0.17	0.17	Z4	0.073	0.062	0.050	0.035	0.040	0.038	0.040
Z6	0.19	0.19	0.20	0.20	0.18	0.18	0.19	Z5	0.064	0.050	0.042	0.025	0.030	0.032	0.033
Z7	0.19	0.19	0.20	0.20	0.17	0.18	0.18	Z8	0.087	0.035	0.025	0.219	0.189	0.006	0.018
Z10	0.17	0.16	0.18	0.17	0.16	0.16	0.16	Z9	0.093	0.040	0.030	0.189	0.168	0.013	0.022
Z11	0.17	0.17	0.18	0.18	0.16	0.16	0.17	Z12	0.049	0.038	0.032	0.006	0.013	0.027	0.026
Z14	0.17	0.17	0.19	0.18	0.16	0.17	0.18	Z13	0.049	0.040	0.033	0.018	0.022	0.026	0.026
TSF-u(PCA) subsample, Group G_1								TSF-u(PCA) subsample, Group G_2							
X	Z2	Z3	Z6	Z7	Z10	Z11	Z14	X	Z1	Z4	Z5	Z8	Z9	Z12	Z13
Z2	0.20	0.19	0.21	0.21	0.18	0.19	0.19	Z1	0.152	0.075	0.066	0.081	0.087	0.051	0.051
Z3	0.19	0.19	0.21	0.20	0.18	0.18	0.19	Z4	0.075	0.066	0.053	0.036	0.041	0.040	0.042
Z6	0.21	0.21	0.22	0.22	0.19	0.20	0.20	Z5	0.066	0.053	0.043	0.026	0.031	0.033	0.034
Z7	0.21	0.20	0.22	0.22	0.19	0.19	0.20	Z8	0.081	0.036	0.026	0.198	0.170	0.007	0.018
Z10	0.18	0.18	0.19	0.19	0.17	0.17	0.18	Z9	0.087	0.041	0.031	0.170	0.150	0.014	0.023
Z11	0.19	0.18	0.20	0.19	0.17	0.18	0.18	Z12	0.051	0.040	0.033	0.007	0.014	0.028	0.027
Z14	0.19	0.19	0.20	0.20	0.18	0.18	0.19	Z13	0.051	0.042	0.034	0.018	0.023	0.027	0.028

Table 1: Variance-covariance matrix, (2), of G_1 and G_2 variables for IBOSS subdata, original data and TSF-u(PCA) subsample

3.3.1 Griewangk's function for response variable

To generate a new response $y = f(\mathbf{x}) = f(x_1, x_2, \cdots, x_d)$, we modified the Griewangk's function which has many widespread local minima regularly distributed. For this and many other test functions, see `http://www.sfu.ca/~ssurjano/`. As the input variables' range in the original function was assumed to be within (-600, 600), which is not consistent with the range of Chemical Sensors Data, we had to make a slight generalization, resulting in

$$f(\mathbf{x}) = \sum_{i=1}^{d} \frac{(x_i - a_i)^2}{b_i} + \sigma \times \prod_{i=1}^{d} \cos\left(\frac{c_i x_i}{\sqrt{i}}\right) + e \tag{14}$$

with possible parameters that can be selected. The original function is a special case with $a_i = 0$, $b_i = 4000$, $c_i = 1$ for all $i = 1, 2, \cdots, d$, $\sigma = 1$ and $e = 0$. For our simulation, we choose $d = 4$ dimensions for input variables (to be selected) from Z_i with $i = 1, 2, \cdots, d$,

1. $(a_1, a_2, \cdots, a_d)$ should be near the minimal point, and we choose $a_i = 3$, all i.
2. $(b_1, b_2, \cdots, b_d)$, controls the scale of d-input variables, we choose $b_i = 4000$, all i.
3. $(c_1, c_2, \cdots, c_d)$ controls the "degree of fluctuation" of the cosine function, $c_i = 5$ for all i.
4. σ controls the relative weight of the product term to the function $f(\mathbf{x})$ and $\sigma = 30$.
5. e is a constant. Clearly, if we choose $e = \sigma$, then it will make $f(\mathbf{x}) \geq 0$.

3.3.2 Inputs for Griewangk's function and covariates for GAM

Since the effective PCA dimension is 4 for the original data, we use $d = 4$ input variables selected from $Z_1, Z_2, \cdots, Z_{14}$ to simulate the Griewangk's function in (14) and choose 'a set of covariates (say, also 4 variables) to be used for building GAM based on the training sample (IBOSS or TSF-u(PCA)). We evaluate and compare two possible set selections chosen from two groups as in (3), with high pairwise correlation and (4) with low pairwise correlation . Specifically, we choose four inputs variables from two groups

$$B_1 = \left\{Z_1, Z_4, Z_8, Z_{12}\right\} \tag{15}$$

and

$$B_2 = \left\{ Z_6, Z_7, Z_{10}, Z_{11} \right\} \tag{16}$$

to train and build the GAM with both training samples. To simulate the response using Griewangk's function, we choose two input variables from each of the two groups not selected as covariates

$$B_G = \left\{ Z_2, Z_3, Z_5, Z_9 \right\}. \tag{17}$$

Intuitively, one would expect to build a better GAM with covariates with no apparent linear relationship like covariates in B_2. We choose covariates from B_G having no duplications with both B_1 and B_2 to produce Griewangk's function to minimize the possible confounding effect of using the same variable for response and covariates.

3.4 Empirical evaluation on simulated Chemical Sensors Data

With the simulated response for the Chemical Sensors Simulated Data, we evaluate the performance of building two GAMs based on two different training samples from IBOSS subdata and our proposed sampling method. Their empirical results comparing the two predicted responses (IBOSS vs. TSF-u(PCA)) on two samples (training samples vs. test samples) with two different GAM covariates are shown in Figures 7 and 8. To separate various data points, we use the following gray-scale schemes:

1. light gray for the original simulated data points,
2. dark gray for the training data of 280 points and test data of 10,000 points,
3. black for the predictions on the training and the test data points,
4. "□" for the predicted values based on GAM using IBOSS subdata, and "+" for the predicted values based on GAM using TSF-u(PCA).

Since it is infeasible to plot high-dimensional data, we project the response Y and its predicted response $\hat{Y}$ on its input variables $Z_i, i = 1, 2, \cdots, 14$. To reduce the number of plots, we choose only projection on Z_1, Z_2, Z_4, Z_8.

There are two main objectives in this empirical study:

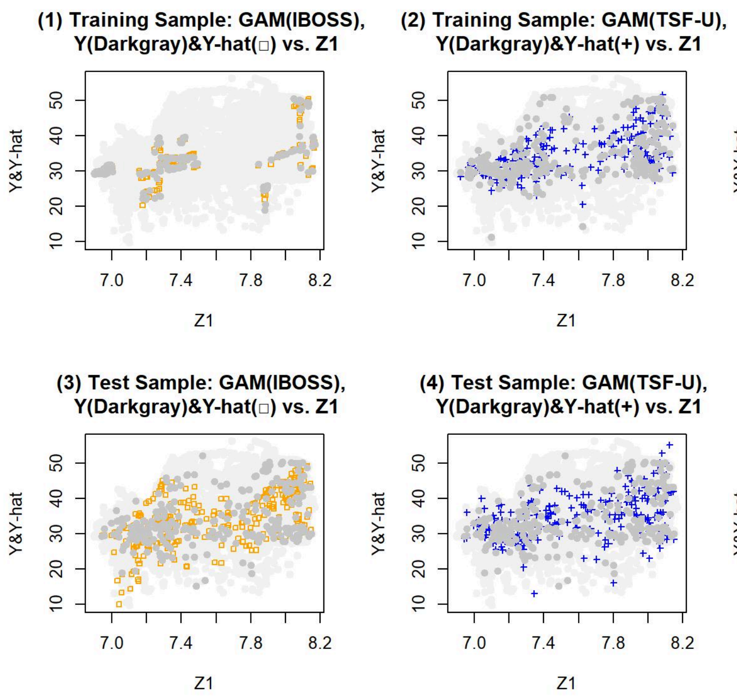
1. We demonstrate the importance of choosing a good training sample (via the proposed method) for the purpose of the model building via GAM. As expected, building GAM with IBOSS subdata performed poorly, especially in the test sample. In contrast, our proposed method can yield a training sample that will be evenly spaced out over the data dimension space and have a similar distribution as that of a (randomly selected) test sample. Therefore, building a GAM with such a training sample would have similar accuracy as the training data.

2. We evaluate the effect of choosing covariates for the GAM. We show that GAM is quite "robust" in the sense that it does not require the same variables used in the response function. Based on the empirical study, we find that it would be better to choose covariates that are not highly correlated.

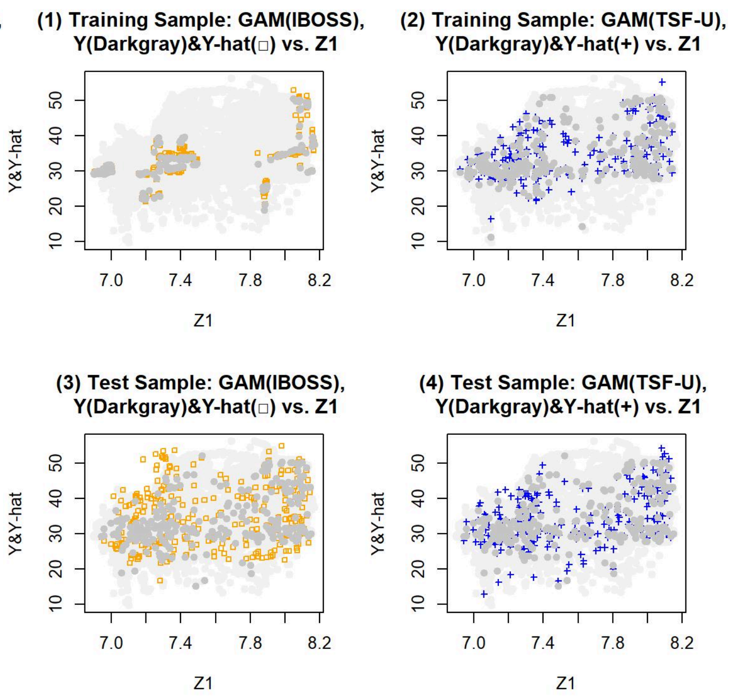
Listed below are some key findings from the plots:

1. As expected, the subdata from IBOSS procedure failed to cover evenly over the data space and it is unsuitable as the training sample for a complicated response function. The plots of the IBOSS procedure show GAM can fit well for the training data reasonably well but it failed on randomly selected test samples. In contrast, our proposed method, TSF-u(PCA), will choose more uniformly spaced points over the data space of X dimensions or its PCA dimensions (of much reduced dimensions). Consequently, the training sample based on the proposed sampling method would be more likely to capture the actual shape of the complicated response function.

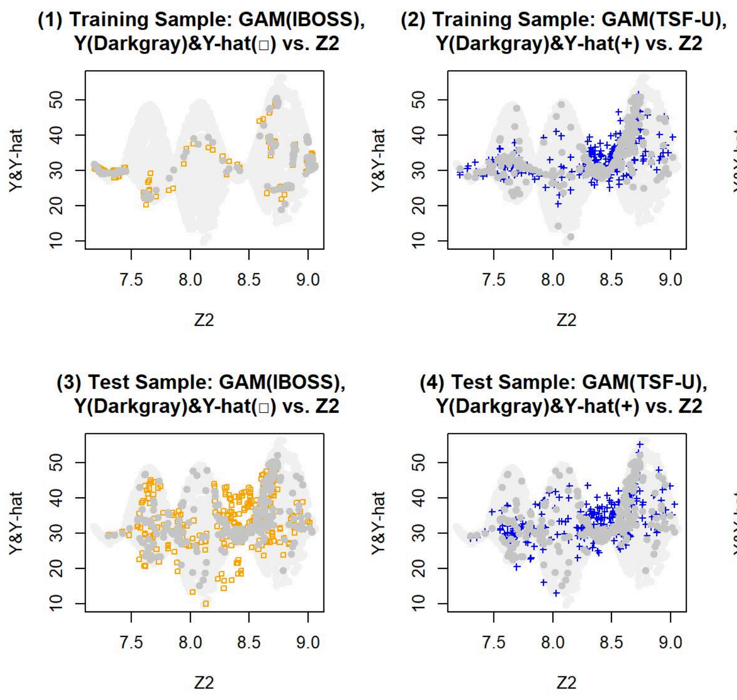
2. Choosing covariates for building GAM is also important. It appears that choosing "more correlated" variables (e.g. Z_1, Z_4, Z_8, Z_{12}) as covariates for GAM is slightly better then choosing "more uncorrelated" variables (e.g. Z_6, Z_7, Z_{10}, Z_{11}) as covariates for GAM. It is interesting to investigate further. Our empirical study indicates that GAM is fairly "robust" in terms of the possible choices of covariates. This is important because we do not need to choose the "correct" covariates to build GAM, as long as we have a "good" training sample.



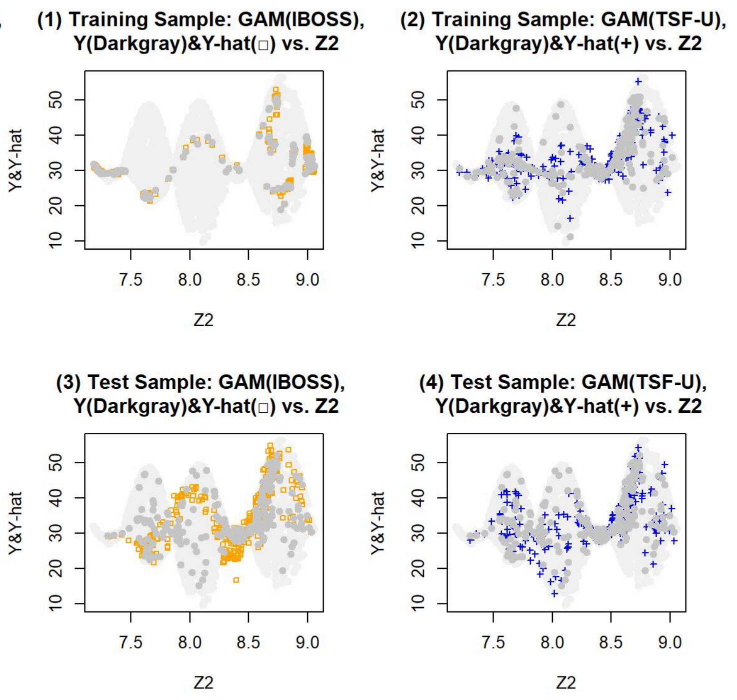
(a) $\hat{Y} = \hat{f}(Z_1, Z_4, Z_8, Z_{12})$ vs. Z_1



(b) $\hat{Y} = \hat{f}(Z_6, Z_7, Z_{10}, Z_{11})$ vs. Z_1

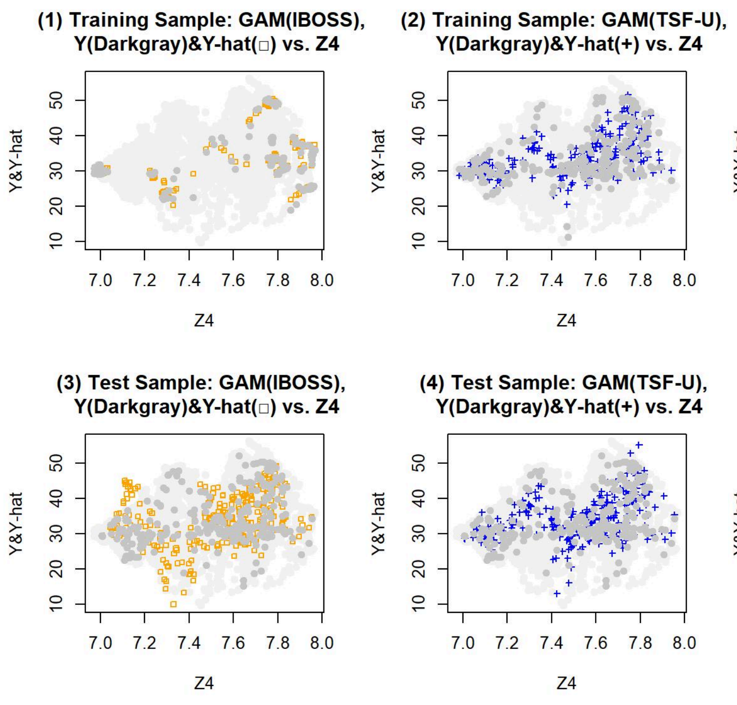


(c) $\hat{Y} = \hat{f}(Z_1, Z_4, Z_8, Z_{12})$ vs. Z_2

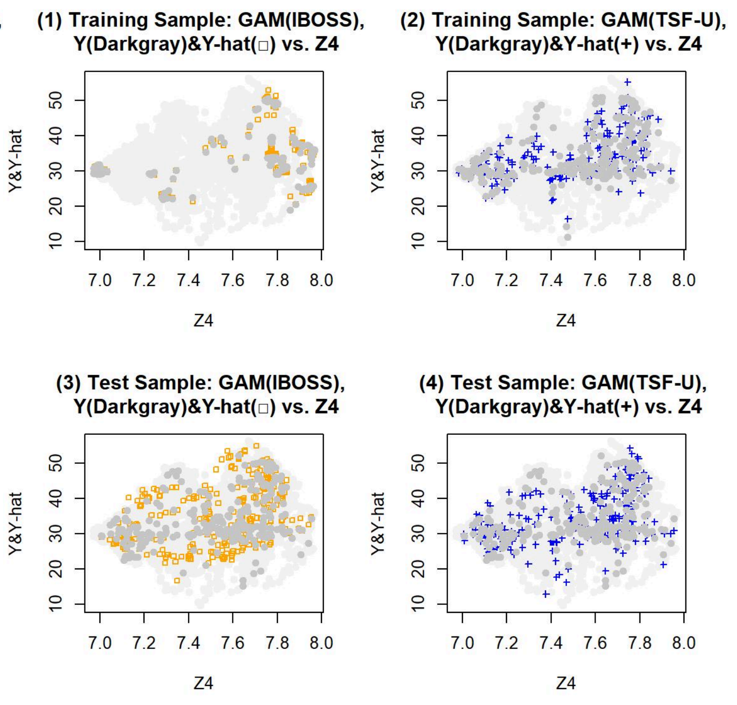


(d) $\hat{Y} = \hat{f}(Z_6, Z_7, Z_{10}, Z_{11})$ vs. Z_2

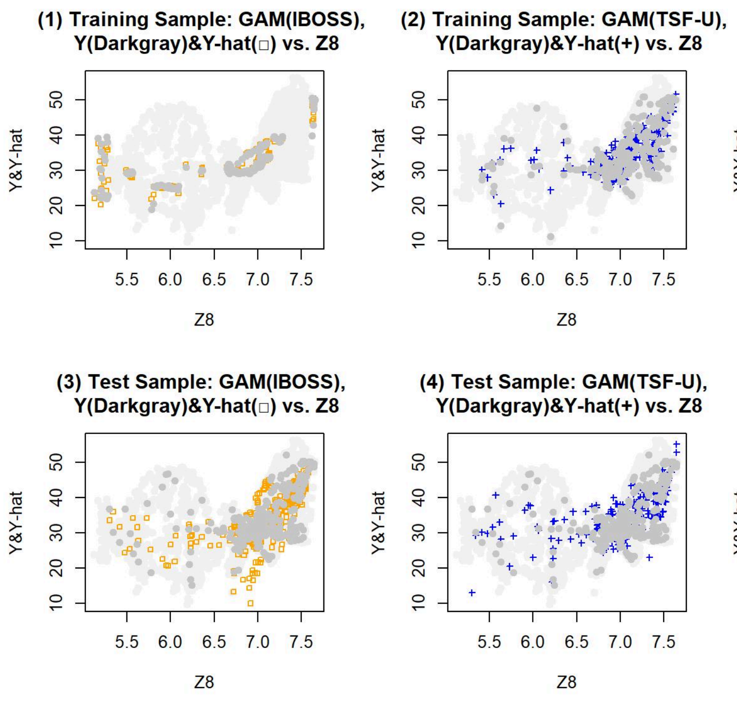
Figure 7: $Y = f(Z_2, Z_3, Z_5, Z_9)$, Y and $\hat{Y}$ vs. Z_1 and Z_2



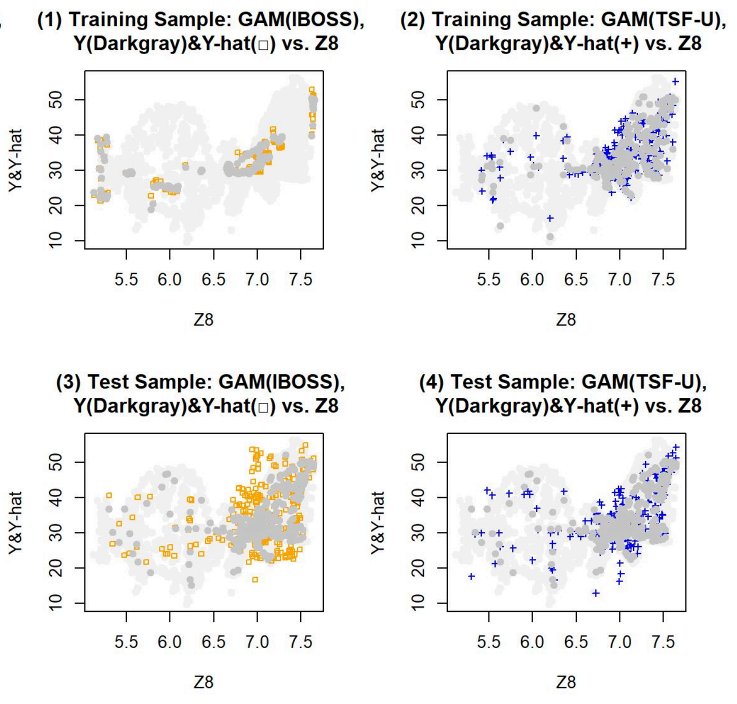
(a) $\hat{Y} = \hat{f}(Z_1, Z_4, Z_8, Z_{12})$ vs. Z_4



(b) $\hat{Y} = \hat{f}(Z_6, Z_7, Z_{10}, Z_{11})$ vs. Z_4



(c) $\hat{Y} = \hat{f}(Z_1, Z_4, Z_8, Z_{12})$ vs. Z_8



(d) $\hat{Y} = \hat{f}(Z_6, Z_7, Z_{10}, Z_{11})$ vs. Z_8

Figure 8: $Y = f(Z_2, Z_3, Z_5, Z_9)$, Y and $\hat{Y}$ vs. Z_4 and Z_8

4 Summary and concluding remarks

To effectively analyze big datasets, it is important to take certain steps to create a useful model for forecasting. It is ideal (1) to select the training subdata carefully so that it well represents the full data; (2) to apply appropriate dimension reduction (e.g., PCA) to reduce the number of possible features to a manageable size; (3) to avoid the potential issue of over-fitting with a huge number of input variables; and (4) to build a useful modeling procedure with the help of dimension reduction and a good sampling scheme to select a training subdata.

Specifically, IBOSS makes a not-so-realistic assumption that the linear model is the appropriate model, and the subdata selected through IBOSS is inappropriate for fitting more complicated models (e.g., GAM). On the contrary, the proposed method TSF-u(PCA) is able to select a good training sample, which is the key to building a good prediction model that would perform well in both the training sample and the test sample.

It is shown that TSF-u(PCA) preserves key characteristics of the full data, such as the variance-covariance matrix and space-filling properties. Furthermore, TSF-u(PCA) is more computationally efficient than IBOSS, while the design efficiency of TSF-u(PCA) is comparable to that of IBOSS. Most importantly, the models based on TSF-u(PCA) can maintain consistent and high prediction accuracy on both training and test data. Given the space limitation, we have a limited comparison of the performance of the proposed method, outside of timing, via simulation. Further extensive evaluations and more comprehensive comparisons between IBOSS and TSF-u(PCA) can be studied as well.

5 Supplementary Materials

R Markdown for Empirical Evaluation and Simulation: The R markdown "Eva_Sim.Rmd" contains R codes to perform the empirical evaluations and simulations in Section 3. The file also contains codes to load the datasets used as examples in the article.

Conflict of Interest

The authors declare that they have no known competing interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Abdi, H. and Williams, L. J. (2010), "Principal component analysis," *Wiley Interdisciplinary Reviews: Computational Statistics*, 2, 433–459.

Ai, M., Wang, F., Yu, J., and Zhang, H. (2021), "Optimal subsampling for large-scale quantile regression," *Journal of Complexity*, 62, 101512.

Anderson, T. W. (2003), *An Introduction to Multivariate Statistical Analysis*, 3rd ed., John Wiley & Sons.

Cheng, Q., Wang, H. Y., and Yang, M. (2020), "Information-based optimal subdata selection for big data logistic regression," *Journal of Statistical Planning and Inference*, 209, 112–122.

Cochran, W. G. (1977), *Sampling techniques*, 3rd ed., John Wiley & Sons.

Fang, K. T., Lin, D. K. J., Winker, P., and Zhang, Y. (2000), "Uniform design: theory and applications," *Technometrics*, 42, 237–248.

Fonollosa, J., Sheik, S., Huerta, R., and Marco, S. (2015), "Reservoir computing compensates slow response of chemosensor arrays exposed to fast varying gas concentrations in continuous monitoring," *Sensors and Actuators B: Chemical*, 215, 618–629.

Furrer, R., Genton, M. G., and Nychka, D. (2006), "Covariance tapering for interpolation of large spatial datasets," *Journal of Computational and Graphical Statistics*, 15, 502–523.

Gramacy, R. B. and Apley, D. W. (2015), "Local gaussian process approximation for large computer experiments," *Journal of Computational and Graphical Statistics*, 24, 561–578.

Hastie, T. J. and Tibshirani, R. J. (1986), "Generalized Additive Models," *Statist. Sci.*, 43, 297–310.

— (1990), *Generalized additive models*, vol. 43, CRC press.

Hoare, C. A. (1961), "Algorithm 64: Quicksort," *Communications of the ACM*, 4, 321.

Joseph, V. R. and Mak, S. (2021), "Supervised compression of big data," *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 14, 217–229.

Li, R., Lin, D. K., and Li, B. (2013), "Statistical inference in massive data sets," *Applied Stochastic Models in Business and Industry*, 29, 399–409.

Liu, R. Y., Parelius, J. M., and Singh, K. (1999), "Multivariate analysis by data depth: descriptive statistics, graphics and inference, (with discussion and a rejoinder by Liu and Singh)," *The Annals of Statistics*, 27, 783 – 858.

Mahalanobis, P. C. (1936), "On the generalised distance in statistics," *Proceedings of the National Institute of Sciences of India*, 2, 49 – 55.

Mak, S. and Joseph, V. R. (2018), "Support points," *The Annals of Statistics*, 46, 2562–2592.

Martinez, C. (2004), "Partial quicksort," in *Proceeding of the 6th ACM-SIAM Workshop on Algorithm Engineering and Experiments and 1st ACM-SIAM Workshop on Analytic Algorithms and Combinatorics*, vol. 4, pp. 224–228.

Wang, H., Yang, M., and Stufken, J. (2019), "Information-based optimal subdata selection for big data linear regression," *Journal of the American Statistical Association*, 114, 393–405.

Wang, L., Elmstedt, J., Wong, W. K., and Xu, H. (2021), "Orthogonal subsampling for big data linear regression," *The Annals of Applied Statistics*, 15, 1273 – 1290.