Hard-threshold-based Feature Screening for Ultrahigh-dimensional GLMs: Methods and Software

by

Qianxiang Zang

A thesis submitted to the University of Ottawa

in partial fulfillment of the thesis requirement for the degree of

Doctor of Philosophy in Mathematics and Statistics[1]

Department of Mathematics and Statistics

Faculty of Science

University of Ottawa

[1]The Ph.D. program is a joint program with Carleton University, administered by the Ottawa-Carleton Institute of Mathematics and Statistics

Abstract

In modern scientific research, the dimensionality and complexity of datasets are growing exponentially. Geneticists analyze data from millions of Single Nucleotide Polymorphisms (SNPs) to identify disease associations, while cybersecurity experts monitor vast amounts of data packets in real-time to detect spam and viruses. Financial analysts continually update bankruptcy predictions as new data become available, and social media platforms identify real-time trends and hot topics from immense streams of content. Undoubtedly, these new types of datasets hold great potential for uncovering subtle patterns. However, their ultrahigh dimensionality, complex structures, and the need for real-time processing present considerable challenges for traditional statistical methods. This calls for the development of novel tools that make modern data analytics viable.

To ease the analytical difficulties, it is often beneficial to pre-process a high-dimensional dataset by efficiently detecting and eliminating a large number of features that are irrelevant to the analysis. This strategy is referred to as feature screening, which aims to bring down the computational cost without loss of key information. Among the existing screening techniques, the hard-threshold-based method has attracted a great deal of attention for its high accuracy and stability. **This dissertation develops user-friendly statistical software for this attractive technique and further designs new screening approaches to extend its applicability.**

Specifically, the dissertation consists of the following three self-contained research projects. The *first project* is on developing a publicly available R package **SMLE** for joint feature screening in ultrahigh-dimensional generalized linear models. The package provides a user-

friendly environment to carry out the Sparsity-restricted Maximum Likelihood Estimation (SMLE) screening method, which is computationally convenient and practically effective. In the *second project*, I design a stability-enhanced screening procedure via an iterative splicing for sparse estimation (ISSE) technique. Compared with SMLE, ISSE is insensitive to the initial model and significantly improves the screening accuracy for data with complex correlation structures. In the *third project*, I address feature screening in a streaming-data setup, where batches of new features arrive over time. For this challenging task, I propose a novel Batch Adapted Neighbour Searching (BANS) algorithm, which screens features in real-time based on a one-step approximated hard-thresholding procedure.

This dissertation presents innovative attempts to address the challenges in analyzing ultrahigh dimensional data. The developed software **SMLE** provides effective and publicly available tools of feature screening for researchers from various domains. It has had over **35,000 downloads** from users around the world and has been highlighted in the Canadian Statistical Society's quarterly newsletter in 2021. The two new algorithms ISSE and BANS significantly improves the original SMLE method and substantially extends its application scope to the emerging fields.

Acknowledgements

I would like to take this moment to express my heartfelt gratitude to everyone who supported me throughout these challenging six years of my Ph.D. journey at the University of Ottawa.

First and foremost, I am deeply thankful to the University of Ottawa for the invaluable opportunity to pursue my degree and for the scholarships that made this journey possible. My profound gratitude goes to my supervisors, Dr. Chen Xu and Dr. Kelly Burkett, for their steadfast patience and insightful guidance throughout my studies and research. Your passion for statistics, extensive knowledge, and dedication to productivity have been a continual source of inspiration, motivating me to learn more and strive for excellence in hopes of following in your footsteps. I am also sincerely grateful for the financial support you provided, which was essential to my studies.

I extend special thanks to my lab colleagues, Xingxiang Li, Kaili Jing, and Wei Dong, from whom I have learned so much through our collaborations and discussions.

This research was made possible using the data/biospecimens collected by the Canadian Longitudinal Study on Aging (CLSA). Funding for the Canadian Longitudinal Study on Aging (CLSA) is provided by the Government of Canada through the Canadian Institutes of Health Research (CIHR) under grant reference: LSA 94473 and the Canada Foundation for Innovation, as well as the following provinces, Newfoundland, Nova Scotia, Quebec, Ontario, Manitoba, Alberta, and British Columbia. This research has been conducted using the CLSA Comprehensive Baseline Dataset version 7.0 and the the CLSA Metabolomics dataset version 1.0 under Application Number 180911. The CLSA is led by Parminder

Raina, Christina Wolfson, and Susan Kirkland. The opinions expressed in this article are the authors own and do not reflect the views of the Canadian Longitudinal Study on Aging.

Dedication

This thesis is dedicated to my beloved parents, Meijing Shi and Qizhen Zang. I am deeply grateful for your unwavering support throughout my challenging graduate journey abroad.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Overview

Technological innovations have made a profound impact on the process of knowledge discovery. It is now feasible to collect data of unprecedented size and dimensionality in diverse areas of scientific research. For example, in the healthcare industry, Big Data has revolutionized patient care and medical research [23, 37, 46]. Predictive analytics driven by Big Data can forecast disease outbreaks and patient admissions, allowing hospitals to allocate resources more efficiently and improve patient outcomes. Predictive models can anticipate surges in emergency room visits, enabling hospitals to adjust staffing levels, which leads to better patient care by reducing wait times and ensuring timely treatments [11]. The Genome-Wide Association Study (GWAS), which involves genotyping many individuals at hundreds of thousands to a million genetic markers, has proved useful in discovering the relationship between common variants and complex diseases. By analyzing genetic

data from large populations, GWAS has identified numerous genetic variants associated with diseases such as diabetes, cancer, and heart disease, providing valuable insights into the genetic basis of these conditions and paving the way for personalized medicine [60]. Social media platforms like Twitter and Facebook identify real-time trends and hot topics through endless content streams. A hot topic on Twitter triggers a surge of relevant tweets within a short period, often reflecting events of mass interest. Twitter for early detection of hot topics has become an interesting research problem with immense practical value. Ultrahigh-dimensional data refers to datasets where the number of features (p) greatly exceeds the number of observations (n), commonly expressed as $p \gg n$. The theoretical definition characterizes ultrahigh-dimensional data as datasets in which p increases at an exponential rate relative to n. While earlier literature emphasized the challenges associated with analyzing such data, high-dimensional analysis is now routine across many disciplines. Nevertheless, the term remains relevant when highlighting statistical and computational complexities associated with high-dimensional inference, feature selection, and model interpretability.

While Big Data bring rich resources that enable better prediction models, they seriously challenge conventional analytical methods in computational efficiency, statistical accuracy, and algorithmic stability. In statistical modeling, one particular challenge is that Big Data often contains information collected on a huge number of features, which can be much larger than the sample size; in some applications, the number of features of a dataset (dimensionality) can even be too large to be stored on a single computer. A statistical model with too many features not only brings up a high fitting cost, but it often leads to low predictive accuracy and poor model interpretability. Developing viable and effective

tools to process ultrahigh-dimensional data has been a focus in modern statistics and related fields.

While an analyst may include a large number of features at the initial stage of modeling, it is often found that only a small number of them are useful in subsequent analyses. For example, in Genome-Wide Association Studies (GWAS), researchers might find only a handful of independent genetic variants reach statistical significance out of hundreds of thousands tested. [60]. Similarly, in text mining and natural language processing, predictive accuracy often depends on a relatively small set of informative keywords or phrases [58]. These observations support the sparsity assumption, where only a small number of features are considered relevant to the outcome of interest, while most features carry negligible or no predictive value. In regression analysis, this sparsity assumption translates to the belief that most regression coefficients are zero, implying that the corresponding features are irrelevant for predictive modeling. Practically, one common strategy for simplifying ultrahigh-dimensional analyses is to first detect and eliminate irrelevant features prior to conducting a detailed analysis on the reduced subset. This process, known as feature screening, has attracted considerable attention over the past decade. By effectively reducing dimensionality from ultrahigh to low, feature screening significantly mitigates computational complexity and analytical challenges.

In the literature, there has been vast research on feature screening (see, for example, [12,15,16,30,62,88,89]). Among the available tools, the hard-threshold-based screening techniques have gained considerable popularity for their methodological simplicity, statistical accuracy, and practical reliability. This dissertation addresses new research problems for feature screening arising from different applications of the hard-threshold-based tech-

niques. The approaches developed for this dissertation are innovative attempts to address the challenges in analyzing ultrahigh-dimensional data.

1.2 Review of feature screening

Feature screening is a popular tool in the analysis of ultrahigh-dimensional data. Different from the goal of accurately selecting features, it serves as a pre-processing technique that aims to facilitate subsequent analyses by eliminating redundant information from a large feature space.

To address different types of analysis, a variety of screening methods have been proposed in the past decade. These methods mainly fall within two categories: marginal screening and joint screening. The former category ranks and screens features based on their marginal effects on the response; the latter category retains a key set of features by considering their joint effects. In this section, I provide a brief introduction of the selected screening methods in both categories.

To aid in the description, I first provide notation that will apply throughout this chapter. Let $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ be an independent and identically distributed sample collected from a population $\{\boldsymbol{x}, y\}$, where $\boldsymbol{x} = (x_1, \ldots, x_p)^{\mathrm{T}}$ is a vector of p features, and y is a real-valued response variable. Let $\boldsymbol{Y} = (y_1, \ldots, y_n)^{\mathrm{T}}$ be the response vector and $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)^{\mathrm{T}}$ be the $n \times p$ feature matrix. For any index set $\mathcal{M} \subset \{1, \ldots p\}$, $\boldsymbol{X}_{\mathcal{M}}$ denotes the sub-matrix of $\boldsymbol{X}$ with columns specified by $\mathcal{M}$. Similarly, for any vector v, $v_{\mathcal{M}}$ denotes the sub-vector of v. It is assumed that only a small number of features influence y and are thus relevant to the analysis. Denote by $\mathcal{M}_*$ the ground truth or the true collection of all relevant features;

the goal of feature screening is to eliminate most irrelevant features in $\{1 \dots p\} \backslash \mathcal{M}_*$.

1.2.1 Marginal screening methods

Sure Independence Screening

The Sure Independence Screening (SIS) method, introduced by Fan and Lv [13], is a feature screening technique tailored for the linear model. Suppose that

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \epsilon,$$

where $\boldsymbol{\beta} = (\beta_1, \dots, \beta_p)^\top$ is a $p \times 1$ vector of regression coefficients and ϵ is random noise. The true regression coefficient vector, denoted as $\boldsymbol{\beta}^* = (\beta_1^*, \dots, \beta_p^*)^\top$, is assumed to be sparse, meaning that only a small subset of its elements are nonzero. Specifically, we define the set of relevant predictors as:

$$\mathcal{M}_* = \{1 \leq j \leq p : \beta_j^* \neq 0\}.$$

Under this sparsity assumption, most of the coefficients in $\boldsymbol{\beta}^*$ are zero, reflecting the belief that the majority of the features do not have meaningful predictive relationships with the response variable $\boldsymbol{Y}$.

The idea of SIS is very straightforward: it ranks all p features based on their marginal effects on the response, and screens out features with weak marginal effects. Specifically, let $\boldsymbol{X}_j$ denote the jth column of $\boldsymbol{X}$ and assume that $\boldsymbol{X}_j$ for $1 \leq j \leq p$ has been standardized with zero sample mean and unit sample standard deviation. The SIS method computes a

screening utility

$$\widehat{\omega}_j = \frac{1}{n}\boldsymbol{X}_j^\top \boldsymbol{Y}$$

for each feature $\boldsymbol{x}_j$ with $j = 1, \ldots, p$. The screening utility $\widehat{\omega}_j$ is the sample Pearson correlation between $\boldsymbol{x}_j$ and $\boldsymbol{y}$; therefore, it quantifies the relevance of $\boldsymbol{x}_j$ through its linear association with the response variable. SIS then ranks all features based on $|\widehat{\omega}_j|$ and screens out features with small $|\widehat{\omega}_j|$ values. With a user-specified screening size $s < n$, which is fixed. SIS retains only a set of top-ranked features in

$$\widehat{\mathcal{M}}_s = \{1 \leq j \leq p : |\widehat{\omega}_j| \text{ is among the } s \text{ largest}\}. \tag{1.1}$$

The SIS procedure is referred to as "sure independence screening" because, under certain conditions such as exponential growth of features relative to sample size, normally distributed errors, sufficiently large signal strength for relevant variables, and controlled correlations among features, it ensures that all the relevant features are included in $\widehat{\mathcal{M}}_s$ with probability approaching one; that is,

$$P(\mathcal{M}_* \subseteq \widehat{\mathcal{M}}) \to 1,$$

as $n \to \infty$ even when $p \gg n$. This property is referred to as the **Sure Screening** property.

Extensions of SIS

Since the seminal work of [13], the SIS framework has been extended to various model settings. For example, Gorst-Rasmussen and Scheike [22] introduced the FAST-SIS method tailored for survival data. They demonstrated that FAST-SIS could maintain the formal

sure screening property within certain single-index hazard rate models. Wu and Yin [70] developed a conditional quantile screening approach designed to identify features that influence the conditional quantile of the response variable given the covariates. This method effectively addresses censored data by employing a weighting scheme that appropriately redistributes the mass to the right, ensuring robustness. They also proved the sure screening properties for scenarios involving both complete and censored response data.

In the same spirit, a series of model-free screening methods have also been developed. Zhu et al. [88] introduced a feature screening procedure under a unified model framework that encompasses a broad spectrum of commonly used parametric and semiparametric models. This method is advantageous when there is limited information about the true underlying model. By not requiring a specific model structure for the regression functions, their approach ensures consistency in ranking features, which aids in achieving overall consistency. Li et al. [31] developed a sure independence screening procedure based on distance correlation (DC-SIS). They showed that the implementation of the DC-SIS does not require model specification (e.g., linear model or generalized linear model) for responses or features and established the sure screening property for the DC-SIS.

1.2.2 Joint screening methods

While marginal screening methods such as SIS are effective in reducing dimensionality, they may miss features that have weak individual effects but significant joint or interaction effects. For example, gene-environment interactions are common in GWAS, where genetic variants alone show negligible predictive power, but when combined with environmental

exposures (e.g., smoking or dietary habits), their collective effect on disease risk becomes substantial [26]. Recognizing these limitations, researchers have proposed enhanced screening techniques that explicitly account for joint or interaction effects among features, aiming to retain variables with critical but subtle influences that standard marginal methods might otherwise overlook.

Iterative Sure Independence Screening

As a marginal screening method, it is well-known that SIS faces challenges under two types of joint effects. In the first scenario, SIS will screen out an important feature when individually it is unrelated to the response but, when considered with other features, the group is collectively relevant to the response variable. In the second scenario, a feature with a weak overall predictive effect may be incorrectly prioritized due to having a higher marginal correlation with the response than other features that are more relevant when considered jointly. To address these issues, Fan and Lv [13] introduced an iterative version of SIS called Iterative Sure Independence Screening (ISIS) by iteratively replacing the response with the residual obtained from the regression of the response on selected covariates in the previous step. By introducing residuals, the influence of selected covariates is diminished, allowing the screening process to focus more sharply on the joint effect between features. This adjustment ensures that the model prioritizes features with joint impact, enhancing its accuracy and efficiency.

Specifically, ISIS applies SIS to select a set $\widehat{\mathcal{A}}_1$ containing k_1 indices in the first step and then employs a penalized likelihood method to refine this set into a subset $\widehat{\mathcal{M}}_1$, which serves as a preliminary estimate of important feature indices.

Next, for each covariate $j \in \widehat{M}^c = \{1, \ldots, p\} \setminus \widehat{\mathcal{M}}_1$ and the current coefficient estimate

$\widehat{\boldsymbol{\beta}}$, they compute:

$$\tilde{\omega}_j = N^{-1} \sum_{i=1}^{N} L(y_i, \boldsymbol{x}_{i,\widehat{\mathcal{M}}_1}^{\top} \widehat{\boldsymbol{\beta}}_{\widehat{\mathcal{M}}_1} + x_{ij} \widehat{\boldsymbol{\beta}}_j),$$

where L is the log-likelihood function and $\boldsymbol{x}_{i,\widehat{\mathcal{M}}_1}$ serves as sub-vector of $\boldsymbol{x}_i$ from index set $\widehat{\mathcal{M}}_1$. Here, $\tilde{\omega}_j$ quantifies the additional contribution of the j-th feature, considering the presence of features in $\widehat{\mathcal{M}}_1$. By ordering $\{\tilde{\omega}_j : j \in \widehat{\mathcal{M}}_1^c\}$, a set $\widehat{\mathcal{A}}_2$ is formed, comprising the indices corresponding to the largest k_2 elements. Following this prescreening step, a penalized likelihood method is then applied to the combined set $\widehat{\mathcal{M}}_1 \cup \widehat{\mathcal{A}}_2$, yielding an updated set $\widehat{\mathcal{M}}_2$. This iterative process of adding and removing features continues until a set of indices $\widehat{\mathcal{M}}_l$ is obtained that either meets the desired screening size or remains unchanged $\widehat{\mathcal{M}}_l = \widehat{\mathcal{M}}_{l-1}$.

Forward Regression

Another approach to handle situations where some variables are jointly correlated but marginally uncorrelated with the response is forward regression, which was proposed for feature screening by Wang [62]. This method is a stepwise regression technique that starts with an empty model and incrementally adds variables to the regression model. The objective is to identify a model that best explains the data by including one covariate at a time. The procedure continues until a pre-specified number of steps is reached or all covariates are included. The forward regression algorithm is outlined in Algorithm 1.1 below.

Algorithm 1.1 The forward regression algorithm.

Input: Data $\{\boldsymbol{X}, \boldsymbol{Y}\}$, $\mathcal{F} = \{1, \ldots, p\}$.

Output: A covariates index $\widehat{\mathcal{M}}$.

1: (Initialization). Set $\widehat{\mathcal{M}}^{(0)} = \emptyset$.

2: (Forward regression). At the k-th step ($k \geq 1$), given $\widehat{\mathcal{M}}^{(k-1)}$, for each $j \in \mathcal{F} \setminus \widehat{\mathcal{M}}^{(k-1)}$, construct a candidate model $\mathcal{S}_j^{(k-1)} = \widehat{\mathcal{M}}^{(k-1)} \cup \{j\}$. Compute

$$\mathrm{RSS}_j^{(k-1)} = \boldsymbol{Y}^\top (I - H_j^{(k-1)}) \boldsymbol{Y},$$

where

$$H_j^{(k-1)} = \boldsymbol{X}_{\mathcal{S}_j^{(k-1)}} (\boldsymbol{X}_{\mathcal{S}_j^{(k-1)}}^\top \boldsymbol{X}_{\mathcal{S}_j^{(k-1)}})^{-1} \boldsymbol{X}_{\mathcal{S}_j^{(k-1)}}^\top$$

is a projection matrix and I is the identity matrix.

3: Identify

$$a_k = \arg \min_{j \in \mathcal{F} \setminus \mathcal{S}^{(k-1)}} \mathrm{RSS}_j^{(k-1)}$$

and update $\widehat{\mathcal{M}}^{(k)} = \widehat{\mathcal{M}}^{(k-1)} \cup \{a_k\}$.

4: Repeat steps 2-3 until the size of $\widehat{\mathcal{M}}^{(k)}$ meets the desired screening size or $\widehat{\mathcal{M}}^{(k)}$ remains unchanged.

Wang [62] explored the properties of forward regression in ultrahigh-dimensional settings. However, forward regression can be unstable during the screening process; small changes in the data can lead to significantly different results. This instability arises because once a covariate is added to the feature set at any step, it remains in the final set. Consequently, in complex and large datasets, this approach may result in a locally optimal model rather than a globally optimal one.

High-dimensional Ordinary Least-squares Projection

In contrast to ISIS and Forward Regression, the method of Wang and Leng [66] addressed the joint effect by improving the rank correlation coefficient. Their approach was inspired by the fact that the ridge regression estimate is approximate to $\boldsymbol{X}_j^\top (\boldsymbol{X}\boldsymbol{X}^\top)^{-1} \boldsymbol{Y}$ when

the ridge parameter approaches 0. They proposed a novel and simple screening technique called the High-dimensional Ordinary Least-Squares Projection (HOLP) based on a new rank correlation coefficient $\breve{\omega}_j$:

$$\breve{\omega}_j = \boldsymbol{X}_j^\top (\boldsymbol{X}\boldsymbol{X}^\top)^{-1}\boldsymbol{Y}, \text{ for } j = 1, 2, \ldots, p.$$

HOLP involves ranking all features according to $|\breve{\omega}_j|$ and selecting the top features. Specifically, let s be the number of features retained after screening. They choose a subset of features $\widehat{\mathcal{M}}_s$ as:

$$\widehat{\mathcal{M}}_s = \{j : |\breve{\omega}_j| \text{ are among the largest } s \text{ of all } |\breve{\omega}_j|\}$$

or

$$\widehat{\mathcal{M}}_\gamma = \{j : |\breve{\omega}_j| \geq \gamma\}$$

for some threshold γ. The HOLP approach provides a diagonally dominant projection matrix defined as $\mathbf{X}^\top(\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{X}$. Diagonal dominance implies that the diagonal elements of the projection matrix are significantly larger in magnitude compared to the off-diagonal elements, resulting in reduced correlations between features within this projected space. Because the correlations between features become smaller due to diagonal dominance, the relative ranking of predictor importance remains stable and is less distorted. Thus, HOLP better preserves the rank order of features according to their true predictive influence on the response, even in the presence of correlated predictors. Unlike ISIS and forward regression, which rely on goodness-of-fit measures, HOLP focuses on ranking feature importance

directly. The HOLP estimator can be seen as the other extreme of the ridge regression estimator by letting the ridge parameter approach 0. Wang and Leng [66] also showed that HOLP possesses the sure screening property and gives consistent variable selection without the strong correlation assumption, and has a low computational complexity.

Best-Subset Selection with splicing

Best-subset selection with splicing (BESS) is a recently proposed technique to improve the screening accuracy with a moderate computational cost. It aims to find a small subset of features by splicing subsets of influential and irrelevant covariates, so that the resulting feature set is expected to have the most desirable accuracy while maintaining simplicity and interpretability. The BESS algorithm employs a distinctive feature called "splicing", which significantly reduces the computational complexity involved in best-subset selection. This technique evaluates a finite number of possible subsets at each level, efficiently narrowing down the search space. In addition, BESS uses an information criterion to balance the trade-off between model complexity and goodness-of-fit, ensuring that the estimate associated with thescreening model is accurate.

In the backward splicing step, for each variable $j \in \widehat{\mathcal{M}}$, the authors compute the quantity

$$L_{-j} = \frac{1}{2N} \sum_{i=1}^{N} \left(y_i - \widehat{\beta}_0 - \sum_{k \neq j} x_{ik} \widehat{\beta}_k \right)^2 . \tag{1.2}$$

and then remove the variable with the smallest value of L_{-j}. In the forward splicing step,

for each variable $j \notin \widehat{\mathcal{M}}$, the authors compute the quantity

$$L_{+j} = \frac{1}{2N} \sum_{i=1}^{N} \left(y_i - \widehat{\beta}_0 - \sum_{k \in \widehat{\mathcal{M}}} x_{ik} \widehat{\beta}_k - x_{ij} \beta_j \right)^2 . \tag{1.3}$$

and then add the variable with the largest value of L_{+j}. The details of BESS are summarized in Algorithms 1.2 and 1.3.

Algorithm 1.2 BESS: Best-Subset Selection with a given screening size s.

Input: $\boldsymbol{X}, \boldsymbol{Y}$, a positive integer $k_{\max}$, and a threshold τ_s.
Initialize $\widehat{\mathcal{M}}^0 = \left\{ j : \sum_{i=1}^{n} \left| \left(\frac{\boldsymbol{X}^T \boldsymbol{Y}}{\sqrt{\boldsymbol{X}^T \boldsymbol{X}}} \right)_j \right| \leq \left(\frac{\boldsymbol{X}^T \boldsymbol{Y}}{\sqrt{\boldsymbol{X}^T \boldsymbol{X}}} \right)_{(s)} \right\}$, $Z^0 = (\widehat{\mathcal{M}}^0)^c$, and $\boldsymbol{\beta}^0_{\widehat{\mathcal{M}}^0} = (\boldsymbol{X}^T_{\widehat{\mathcal{M}}^0} \boldsymbol{X}_{\widehat{\mathcal{M}}^0})^{-1} \boldsymbol{X}^T_{\widehat{\mathcal{M}}^0} \boldsymbol{Y}$, $d^0_{\widehat{\mathcal{M}}^0} = \boldsymbol{\beta}^0_{\widehat{\mathcal{M}}^0}$, $\lambda_0 = \frac{\boldsymbol{X}^T (\boldsymbol{Y} - \boldsymbol{X} \boldsymbol{\beta}^0)}{n}$.
for $m = 0, 1, \ldots$ **do**
    $(\boldsymbol{\beta}^{m+1}, d^{m+1}, \widehat{\mathcal{M}}^{m+1}, Z^{m+1}) = \text{Splicing}(\boldsymbol{\beta}^m, d^m, \widehat{\mathcal{M}}^m, Z^m, k_{\max}, \tau_s)$.
    if $(\widehat{\mathcal{M}}^{m+1}, Z^{m+1}) = (\widehat{\mathcal{M}}^m, Z^m)$ **then**
        stop
    end if
end for
Output the screening set $\widehat{\mathcal{M}} = \widehat{\mathcal{M}}^{m+1}$.

Algorithm 1.3 Splicing $(\boldsymbol{\beta}, d, \widehat{\mathcal{M}}, I, k_{\max}, \tau_s)$

Input: $\boldsymbol{\beta}, d, \widehat{\mathcal{M}}, I, k_{\max}$, and τ_s.
Initialize $L_0 = \frac{1}{2n}\|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2$, and set

$$\xi_j = \frac{\boldsymbol{X}_j^T \boldsymbol{X}_j}{2n}(\boldsymbol{\beta}_j)^2, \quad \zeta_j = \frac{\boldsymbol{X}_j^T \boldsymbol{X}_j}{2n}\left(\frac{d_j}{\boldsymbol{X}_j^T \boldsymbol{X}_j/n}\right)^2, \quad j = 1, \ldots, p.$$

for $k = 1, 2, \ldots, k_{\max}$ **do**
    Compute $\widehat{\mathcal{M}}_k = \left\{j \in \widehat{\mathcal{M}} : \sum_{i \in \widehat{\mathcal{M}}} \mathbf{1}(\xi_i \geq \xi_j) \leq k\right\}$ and $I_k = \left\{j \in I : \sum_{i \in I} \mathbf{1}(\zeta_i \leq \zeta_j) \leq k\right\}$.
    Let $\widehat{\mathcal{M}}_{k+1} = (\widehat{\mathcal{M}} \setminus \widehat{\mathcal{M}}_k) \cup I_k$, $I_{k+1} = (I \setminus I_k) \cup \widehat{\mathcal{M}}_k$, and compute

$$\widehat{\beta}_{\widehat{\mathcal{M}}_{k+1}} = (X^T_{\widehat{\mathcal{M}}_{k+1}} X_{\widehat{\mathcal{M}}_{k+1}})^{-1} X^T_{\widehat{\mathcal{M}}_{k+1}} y, \quad \widehat{\beta}_{I_{k+1}} = 0,$$

$$\widehat{d} = \boldsymbol{X}^T(\boldsymbol{Y} - \boldsymbol{X}\widehat{\boldsymbol{\beta}}_{\widehat{\mathcal{M}}_{k+1}}), \quad L_n(\widehat{\beta}_{\widehat{\mathcal{M}}_{k+1}}) = \frac{1}{2n}\|\boldsymbol{Y} - \boldsymbol{X}\widehat{\boldsymbol{\beta}}_{\widehat{\mathcal{M}}_{k+1}}\|_2^2.$$

    if $L_n(\widehat{\beta}_{\widehat{\mathcal{M}}_{k+1}}) > L_n(\widehat{\beta})$ **then**
        $(\boldsymbol{\beta}, \widehat{d}, \widehat{\mathcal{M}}, I) = (\widehat{\boldsymbol{\beta}}, \widehat{d}, \widehat{\mathcal{M}}_{k+1}, I_{k+1})$, $L = L_n(\widehat{\boldsymbol{\beta}}_{\widehat{\mathcal{M}}_{k+1}})$.
    end if
end for
if $L_0 - L < \tau_s$ **then**
    Output $(\widehat{\boldsymbol{\beta}}, \widehat{d}, \widehat{\mathcal{M}}, \widehat{I})$.
end if

The BESS algorithm iterates these steps until the stopping criteria are met. The computational complexity of the ABESS method is polynomial in terms of the sample size n, the dimensionality p, and the squared sparsity level s^2, with additional logarithmic factors involving these terms. In addition, the algorithm identifies the true set of relevant predictors with high probability, providing a reliable approach to feature screening in high-dimensional data analysis.

Hard-threshold-based Feature Screening

Hard-threshold-based screening methods are inspired by optimization problems involving explicit sparsity constraints (called L_0 problems). Solutions to these problems inherently produce sparse coefficient estimates, directly identifying which features are relevant. By setting coefficients below a certain threshold to zero, these methods select only the most important features, making them suitable for feature screening in high-dimensional settings.

To be specific, suppose that $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ are independently collected from model $\mathcal{M}(\boldsymbol{\beta})$, where $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^\top$ is a p-dimensional model coefficient indicating the effects of $(x_1, \ldots, x_p)$ on the response. Suppose that only a small number of features have non-zero effects and thus it is sensible to consider the following estimation problem

$$\min_{\boldsymbol{\beta}} \mathcal{L}_n(\boldsymbol{\beta}), \quad \text{subject to} \quad \|\boldsymbol{\beta}\|_0 \leq s, \tag{1.4}$$

where $\mathcal{L}_n(\cdot)$ is a smooth convex loss under $\mathcal{M}(\boldsymbol{\beta})$ based on the observations $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ and $\|\cdot\|_0$ is the L_0 norm indicating the number of nonzero entries of a vector. By requirement, a solution to (1.4) has at most s non-zero entries, indicating s relevant features; the features with zero effects are then treated as irrelevant ones and are to be screened out. This amounts to identifying the top s features that lead to the minimal loss under the sparsity constraint over model $\mathcal{M}(\boldsymbol{\beta})$. Since the model coefficients are estimated jointly, the joint effects among features are naturally accounted for.

Clearly, finding the global solution to (1.4) is computationally expensive (or even infeasible) when p is huge. However, for feature screening, accurate parameter estimation is not really needed and one viable idea is to obtain a rough solution that (merely) helps to

identify the important features.

To this end, many approximated L_0 algorithms can be applied to feature screening; in particular, the iterative hard-thresholding (IHT) technique has been widely used for its algorithmic simplicity and practical effectiveness [4]. This technique approximates $\mathcal{L}_n(\boldsymbol{\beta})$ by

$$h_n(\boldsymbol{\gamma};\boldsymbol{\beta}) = \mathcal{L}_n(\boldsymbol{\beta}) + (\boldsymbol{\gamma}-\boldsymbol{\beta})^\top \mathcal{L}_n'(\boldsymbol{\beta}) + \frac{u}{2}\|\boldsymbol{\gamma}-\boldsymbol{\beta}\|_2^2, \tag{1.5}$$

with a step parameter $u > 0$, where $\|\cdot\|_2$ denotes the vector L_2 norm (Euclidean norm). The first two terms in (1.5) come from the first order Taylor's expansion of $\mathcal{L}_n(\boldsymbol{\beta})$, while the last term is a quadratic penalty on the distance between $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$. The function $h_n(\boldsymbol{\gamma};\boldsymbol{\beta})$ then serves as a surrogate of $\mathcal{L}_n$ at a given $\boldsymbol{\beta}$.

With an initial value $\boldsymbol{\beta}^{(0)}$, one can conveniently update the estimate of $\boldsymbol{\beta}$ by iteratively solving

$$\boldsymbol{\beta}^{(t+1)} = \arg\min_{\boldsymbol{\gamma}} h_n(\boldsymbol{\gamma};\boldsymbol{\beta}^{(t)}), \text{ subject to } \|\boldsymbol{\gamma}\|_0 \le s. \tag{1.6}$$

Note that $h_n(\boldsymbol{\gamma};\boldsymbol{\beta})$ is quadratic and separable in $\boldsymbol{\gamma}$; this makes solving (1.6) a straightforward job. Specifically, the solution to (1.6) has a closed-form expression

$$\boldsymbol{\beta}^{(t+1)} = H_s[\boldsymbol{\beta}^{(t)} - u^{-1}\mathcal{L}_n'(\boldsymbol{\beta}^{(t)})],$$

where $H_s[\cdot]$ is the hard-threshold operator setting all but the s largest elements (in absolute value) of a vector to zero. IHT essentially converts a p-dimensional non-convex optimization problem into p univariate problems, each of which can be solved explicitly.

Utilizing the IHT technique, Xu and Chen [73] proposed a joint feature screening

method via the sparsity-restricted maximum likelihood estimation (SMLE) for generalized linear models; see Section 2.2.2 for a detailed description about SMLE. Yang et al. [74, 75] extended SMLE to ultrahigh-dimensional Cox's and additive models. Qu et al. [45] studied the IHT-based screening for quantile regression with ultrahigh-dimensional heterogeneous data.

1.3 Contributions

Feature screening has received much attention, and it is applicable to a wide range of statistical models in a variety of scientific areas. In this dissertation, I address new research problems in feature screening arising from several applications of the hard-threshold-based screening techniques. The major research topics and scientific contributions of this dissertation are summarized as follows.

(I) New software for joint feature screening

To provide publicly available tools for feature screening with ultrahigh-dimensional data, in my first project, I develop an R package **SMLE**, which is free for download on CRAN at https://cran.r-project.org/web/packages/SMLE/. The package provides a user-friendly environment to carry out the SMLE screening method under the framework of generalized linear models, which include linear, logistic, and Poisson regression as special cases. **SMLE** implements the IHT algorithm to retain a pre-specified number of key features receiving strong support from the likelihood; in the process, the joint effects among features are naturally incorporated. The **SMLE** package includes functions for conducting SMLE-screening and post-screening selection with popular selection criteria such as AIC, BIC,

and extended BIC. It accommodates both numerical and categorical feature inputs, giving users flexibility in controlling various screening parameters.

In **SMLE**, a package manual is provided to document the details of the contained functions; users can learn the usage of each function by checking its input arguments and output values. Extensive numerical studies are conducted to illustrate the promising performance of the package.

The package has had over **35,000 downloads** from the users around the world. It has been highlighted in the Canadian Statistical Society's quarterly newsletter, Liaison 2021. As the primary developer, I was invited to introduce this package at the 2021 Annual Meeting of the Canadian Statistical Society.

(II) Stability-enhanced iterative joint feature screening

While SMLE provides a viable route for feature screening, it relies on searching an informative set of features within a local region of the initial parameters. Its performance can be unstable for data with complicated correlation structures. This motivates my second project, which designs a stability-enhanced screening procedure via iterative splicing for sparse estimation (ISSE) technique. Unlike SMLE, ISSE does not require a careful specification of the initial model; it simply begins with an empty model and detects a small number of key features by iteratively applying the IHT algorithm over a series of the spliced feature sets. The procedure updates the currently retained features for an improved likelihood until no further gain can be obtained. Numerical examples show that ISSE outperforms the original SMLE by suggesting a set of features with a significantly higher likelihood support. Based on ISSE, an adaptive version is further designed to automatically

determine the appropriate screening size (i.e. number of features to be retained). The effectiveness of the new method is supported using both simulated and real data examples.

(III) Efficient Streaming feature screening

In the third project, I address feature screening in a streaming-data setup, where batches of new features arrive over time. Different from the traditional setup, analysts do not have access to the "full feature space" and real-time screening are needed along the streaming process. This brings significant challenges for feature screening in terms of both accuracy and efficiency. To tackle the challenges, I propose a Batch Adapted Neighbour Searching (BANS) algorithm, which screens features in real-time based on a one-step approximated IHT procedure. Specifically, at timestamp t, BANS efficiently determines whether the current set of key features needs to be updated given a batch of new features; this is achieved by evaluating multiple one-step IHT runs with varying search sizes that likely lead to different combinations of the suggested features at timestamp t. Compared with the existing methods, BANS shows a more reliable screening performance by considering joint effects among features in both the set of current key features and the new batch at timestamp t. The merits of BANS are well observed in the numerical studies in terms of both accuracy and efficiency.

1.4 Organization of the Dissertation

The remainder of this dissertation is organized as follows. Chapter 2 introduces my R package **SMLE**, which conducts efficient joint feature screening for ultrahigh-dimensional

GLMs. Chapter 3 proposes a stability-enhanced screening procedure, which is based on a novel ISSE technique and is insensitive to the initial input. Chapter 4 presents the BANS algorithm for real-time streaming feature screening and shows numerical examples to demonstrate its out-performance. Chapter 5 gives a summary of the dissertation and puts forward a few directions for future research.

Chapters 2-4 are self-contained with chapter-specific notations. A reference list is provided at the end of the dissertation. The package manual of **SMLE** is given in the Appendix.

Chapter 2

SMLE: An R Package for Joint Feature Screening in Ultrahigh-dimensional GLMs

This chapter has been submitted to the Journal of Statistical Software: "SMLE: An R Package for Joint Feature Screening in Ultrahigh-dimensional GLMs" by Q. Zang, C. Xu and K. Burkett. It is currently under revision.

As first author and lead developer, I designed the SMLE R package and completed all of the programming. I also extended the method to categorical features and implemented this extension in the R package. I implemented and summarized all simulation studies and the real data analyses. I also took the lead in drafting the manuscript.

2.1 Introduction

In modern scientific research, it is common to encounter ultrahigh-dimensional datasets with a huge number of features. For example, geneticists often need to measure thousands to hundreds of thousands of genes in the hope of discovering those that influence an observable trait; an Internet firewall may scan millions of keywords on data packets in order to determine their security risk. While ultrahigh-dimensional data bring rich resources to explore many unknown areas, they pose simultaneous challenges of computational cost, statistical accuracy, and algorithmic stability for classic statistical methods [15].

When the number of features is huge, it is often reasonable to assume that only a handful of them are relevant to the analysis. In a regression setting, this amounts to assuming that most predictors in an ultrahigh-dimensional model have no effect on the response (i.e., the regression coefficient is zero). With this sparsity assumption, one natural strategy is to screen most irrelevant features out before a more elaborate analysis is conducted. This pre-processing procedure is referred to as feature screening. With dimensionality reduced from high to low, analytical difficulties can be reduced drastically.

In recent years, much research has been done on feature screening. Fan and Lv [14] proposed to screen features based on their marginal Pearson correlations with the response; they referred to this procedure as sure independence screening (SIS) and justified its theoretical effectiveness for linear models. Fan [17] extended SIS to generalized linear models (GLMs). In the same spirit, Zhu [88] proposed a sure independent ranking and screening (SIRS) based on the conditional distribution of the response given each feature. Li [30] developed a model-free sure independence screening based on the distance corre-

lation (DC-SIS). Wu and Yin [71] proposed a distribution function screening by testing the independence between the response and each feature. Zhou [85] proposed a robust screening method for features containing extreme values. Li [32] proposed a distributed screening framework for the divide-and-conquer setup. The list above is certainly far from complete; readers may refer to Liu [34] for a selective overview of feature screening.

In the literature, most screening methods are developed based on the marginal effects of features on the response. Despite the convenience of implementation, these methods are often found to be unreliable in practice, as the joint effects among features are ignored. Features with significant joint effects but showing weak marginal effects are likely to be wrongly screened out. To tackle this issue, Fan and Lv [14] suggested applying SIS iteratively (ISIS) with a smaller number of features retained in each round. Wang [61] suggested using classic forward regression for screening purposes. These strategies help to incorporate some feature joint effects in the screening process. However, they are usually at a high computational cost, which can be unfavorable in many applications.

Starting from a different angle, Xu and Chen [73] proposed a joint feature screening method via the sparsity restricted maximum likelihood estimator (SMLE). With a L_0 penalty specifying the number of features allowed in the model, the method attempts to roughly estimate a handful of the most significant coefficients from the full model while setting all other coefficients to zero. Since the estimation is carried out on the full model, the resulting sparse estimator readily serves as a feature screener, which naturally takes the joint effects among features into account. The SMLE method can be efficiently implemented by an iterative hard-thresholding algorithm (IHT), which does not involve complex numerical operations such as matrix inversion. Each IHT iteration increases the value of

the sparsity-constrained joint likelihood and thereby provides an improved sparse solution, which eventually leads to a reliable screening result. Xu and Chen [73] further justified the sure screening property of SMLE in ultrahigh-dimensional GLMs and proved the convergence of the IHT algorithm. SMLE has been demonstrated to be an effective tool for feature screening; the method has attracted considerable attention in the literature [45, 74, 75].

In this paper, we provide a publicly available R package **SMLE**, which gives a user-friendly environment to carry out SMLE in ultrahigh-dimensional GLMs including linear, logistic, and Poisson models. The package makes use of the crossprod function to handle ultrahigh-dimensional matrix products and includes a well-tuned main function to efficiently conduct SMLE-screening based on the IHT algorithm. With the package, we are able to repeat the same numerical experiments in Xu and Chen [73] at a significantly reduced time cost in comparison with the original code provided by the authors. In the package, we extend SMLE by permitting both numerical and categorical features in the screening, where the categorical features can be automatically identified and encoded by a user-selected method. Moreover, combined with popular selection criteria such as AIC or (extended) BIC, we propose a SMLE-based selection method, which helps to further identify the relevant features after screening. This post-screening selection method can be conveniently conducted within the main screening function or can be used independently on a user-supplied dataset. We illustrate the usage of **SMLE** via extensive numerical examples. The promising performance of the package is observed in comparison with **SIS** [52] and **VariableScreening** [29], which are the standard R packages for feature screening.

The rest of the chapter is organized as follows. In Section 2.2, we give a brief overview of SMLE and the IHT algorithm. In Section 2.3, we discuss implementation details of

the **SMLE** package. In Section 2.4, we illustrate the usage of the package using extensive numerical examples and compare its performance with a few existing R packages. We conclude this chapter in Section 2.5 with a few remarks.

2.2 The SMLE Method

2.2.1 Notation and Problem setup

Suppose the data $\{(y_i, \boldsymbol{x}_i), i = 1, \ldots, n\}$ are collected independently from $(Y, \boldsymbol{x})$, where Y is a response variable and $\boldsymbol{x} = (x_1, \ldots, x_p)$ is a p-dimensional covariate (feature) vector. We postulate a GLM between Y and $\boldsymbol{x}$ as follows. Conditioning on $\boldsymbol{x}$, the distribution of Y is assumed to belong to an exponential family taking the form

$$f(y; \theta) = \exp(\theta y - b(\theta) + c(y)),$$

where θ is the natural or canonical parameter and $b(.)$, $c(.)$ are two known functions. Under the canonical link, $\boldsymbol{x}$ influences Y in the form of a linear combination

$$\theta = \boldsymbol{x}\boldsymbol{\beta},$$

where $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^T$ is a p-dimensional regression coefficient. Popular GLMs with canonical links include the normal linear model, the logistic model, and the log-linear Poisson model [39].

Under the GLM framework, the effect of each feature x_j on the response Y is char-

acterized by the size of the corresponding regression coefficient β_j. In applications, when the number of features p is large, it is often believed that only a small number of the features in $\boldsymbol{x}$ contribute to the variation in Y, which leads to an idealistic assumption that $\boldsymbol{\beta}$ contains many zero elements. With this sparsity assumption, only features with non-zero coefficients are considered to be relevant in explaining the variation of Y. The goal of feature screening is to identify and remove most of the irrelevant features, so that a more elaborate analysis can be conducted only on the features most likely to be related to Y.

2.2.2 The SMLE-screening and IHT algorithm

The idea of SMLE is simple. When p is overly large and most model coefficients are assumed to be zero, it is probably not wise to estimate the entire $\boldsymbol{\beta}$ from scratch. Instead, it is reasonable to consider just estimating some of the coefficients while setting the others to zero from the beginning. This leads to a sparsity-restricted estimation, which readily serves for feature screening. The procedure is known as a hard-thresholding algorithm because it imposes a hard threshold by selecting only the largest or most significant coefficients, forcing the rest of the coefficients to zero. Unlike soft-thresholding methods, which gradually shrink coefficients toward zero, hard-thresholding strictly eliminates features below a certain threshold, thus enforcing sparsity.

Specifically, under the GLM given in Section 2.2.1, the log-likelihood function of $\boldsymbol{\beta}$ is given by

$$l(\boldsymbol{\beta}) = \sum_{i=1}^{n} [y_i \cdot \boldsymbol{x}_i\boldsymbol{\beta} - b(\boldsymbol{x}_i\boldsymbol{\beta})].$$

With a user-specified sparsity $k < p$, the SMLE estimator is defined by

$$\widehat{\boldsymbol{\beta}}_k = \arg\max_{\boldsymbol{\beta}} l(\boldsymbol{\beta}) \quad \text{subject to} \quad ||\boldsymbol{\beta}||_0 \leq k, \tag{2.1}$$

where $\|.\|_0$ is the vector L_0 norm indicating the number of non-zero elements in that vector. Clearly, $\widehat{\boldsymbol{\beta}}_k$ is designed to set all but the most significant k coefficients to be zero; this amounts to identifying k important features supported most by the joint likelihood. When p is large and k is chosen to be much smaller than p, $\widehat{\boldsymbol{\beta}}_k$ can be viewed as a feature screener, which naturally takes the joint effects among features into account. The idea of SMLE has similarities with the use of L_0-regularized techniques in image processing, where sparsity-constrained least-squares methods are frequently used to construct sparse representations for high-resolution images [5, 10].

While SMLE is conceptually simple, carrying out problem (2.1) can be numerically challenging, as it is a high-dimensional combinatorial optimization. However, since our goal is feature screening, finding the global solution to (2.1) is not a major concern. In fact, it would suffice if we can obtain a good local solution, which helps to retain all relevant features.

In this spirit, Xu and Chen [73] proposed an iterative hard-thresholding algorithm (IHT) to approximately solve (2.1). The idea is as follows. With a $\boldsymbol{\gamma}$ close to $\boldsymbol{\beta}$, one can approximate $l(\boldsymbol{\beta})$ by a surrogate function

$$h(\boldsymbol{\beta}, \boldsymbol{\gamma}) = l(\boldsymbol{\gamma}) + (\boldsymbol{\beta} - \boldsymbol{\gamma})^T l'(\boldsymbol{\gamma}) - (u/2)||\boldsymbol{\beta} - \boldsymbol{\gamma}||_2^2, \tag{2.2}$$

where $l'(\boldsymbol{\gamma}) = \partial l(\boldsymbol{\gamma})/\partial \boldsymbol{\gamma}$, $\|.\|_2$ indicates the L_2 (Euclidean) norm, and $u > 0$ is a scale parameter. The first two terms in (2.2) match the Taylor's expansion of $l(\boldsymbol{\beta})$ at $\boldsymbol{\beta} = \boldsymbol{\gamma}$, and the third term is introduced as a regularization term to enhance the convexity.

The reason for introducing $h(\boldsymbol{\beta}, \boldsymbol{\gamma})$ is that it is separable in the components of $\boldsymbol{\beta}$ and thus serves as a surrogate of $l(\boldsymbol{\beta})$ to conveniently carry out the sparsity-restricted maximization over $\boldsymbol{\beta}$. Specifically, with an initial value $\boldsymbol{\beta}^{(0)}$, we can seek a local solution of (2.1) via the following iterative procedure.

$$\boldsymbol{\beta}^{(t+1)} = \arg\max_{\boldsymbol{\beta}} h(\boldsymbol{\beta}, \boldsymbol{\beta}^{(t)}) \quad \text{subject to} \quad ||\boldsymbol{\beta}||_0 < k. \tag{2.3}$$

Let $\boldsymbol{y} = (y_1, \ldots, y_n)^T$ and $\boldsymbol{X} = (\boldsymbol{x}_i^T, \ldots, \boldsymbol{x}_n^T)^T$. Because of the additivity of $\boldsymbol{\beta}$ in h, the optimization in (2.3) takes a unified form

$$\min_{\boldsymbol{\beta}} \frac{1}{2} \left\| \boldsymbol{\beta} - u^{-1}[u\boldsymbol{\beta}^{(t)} + \boldsymbol{X}^T\boldsymbol{y} - \boldsymbol{X}^T b'(\boldsymbol{X}\boldsymbol{\beta}^{(t)})] \right\|_2^2 \quad \text{subject to } \|\boldsymbol{\beta}\|_0 \leq k, \tag{2.4}$$

where b' is the derivative of the $b(.)$ function depending on the choice of GLM [1]. Obviously, if (2.4) does not come with the sparsity constraint, its solution should be

$$\tilde{\boldsymbol{\beta}}^{(t)} = \boldsymbol{\beta}^{(t)} + u^{-1}\boldsymbol{X}^T[\boldsymbol{y} - b'(\boldsymbol{X}\boldsymbol{\beta}^{(t)})], \tag{2.5}$$

which corresponds to a zero loss in the objective function. Thus, the constrained minimum of (2.4) is achieved by choosing the k largest (in absolute value) components of $\tilde{\boldsymbol{\beta}}^{(t)}$.

[1] For normal linear model, $b'(\theta) = \theta$; for logistic model, $b'(\theta) = \exp\{(\theta)\}/(1 + \exp\{(\theta\})$; for Poisson model, $b'(\theta) = \exp\{(\theta)\}$.

Algorithm 2.1 SMLE-Screening via IHT

Input: Data $(\boldsymbol{y}, \boldsymbol{X})$, screening size k, initial $\boldsymbol{\beta}^{(0)}$, initial step size u_0^{-1}, and decrease rate $\tau \in (0, 1)$.

Initialize: Set $t = 0$.

Repeat until stopping criterion is satisfied:

1. Set $\nu = u_0^{-1}$, $r = 0$.
2. **Repeat until** $l(\tilde{\boldsymbol{\beta}}) \geq l(\boldsymbol{\beta}^{(t)})$:
   (a) Compute $\tilde{\boldsymbol{\beta}} = H_k \left[\boldsymbol{\beta}^{(t)} + \nu \boldsymbol{X}^T \left(\boldsymbol{y} - b'(\boldsymbol{X}\boldsymbol{\beta}^{(t)}) \right) \right]$.
   (b) Update $\nu = \tau\nu$.
   (c) Increment $r = r + 1$.
3. Update $\boldsymbol{\beta}^{(t+1)} = \tilde{\boldsymbol{\beta}}$.
4. Set u-search$^{(t)} = r$.
5. Increment $t = t + 1$.

Output: $\boldsymbol{\beta}^{(t)}$, the number of iterations t, the number of u-search tries in each iteration, and an index set of retained features $\widehat{s} = \{1 \leq j \leq p : \beta_j^{(t)} \neq 0\}$.

Consequently, $\boldsymbol{\beta}^{(t+1)}$ in (2.3) has an explicit expression

$$\boldsymbol{\beta}^{(t+1)} = H_k[\tilde{\boldsymbol{\beta}}^{(t)}], \tag{2.6}$$

where $H_k[\boldsymbol{\beta}]$ is the hard-thresholding operator setting all but the k largest components in $|\boldsymbol{\beta}|$ to zero.

In IHT, u^{-1} serves as a step size, which controls the moving distance from $\boldsymbol{\beta}^{(t)}$ to $\boldsymbol{\beta}^{(t+1)}$.

While a larger u^{-1} often helps to boost the iterations, the procedure may fail to converge when u^{-1} is overly large. Thus, to balance algorithm convergence and iteration efficiency, one may choose to adaptively tune u^{-1} at each step (called u-search). [73] proved that, when u^{-1} is small enough, the IHT procedure leads to a non-decreasing likelihood. In our package, we first initialize u^{-1} with a large value and then adaptively decrease its size by a factor of $\tau \in (0, 1)$ until $l(\boldsymbol{\beta}^{(t+1)}) \geq l(\boldsymbol{\beta}^{(t)})$ is satisfied. This seems to be an effective way for achieving sufficiently fast convergence.

We summarize the SMLE-screening procedure via IHT in Algorithm 2.1. As can be seen from the algorithm summary, the procedure involves only simple numerical operations and adaptively tuning u^{-1} often requires only a few tries to succeed. At each iteration, the joint information carried in $\boldsymbol{X}$ is naturally accounted for as a basis for the next update. These merits make SMLE-screening attractive in ultrahigh-dimensional data analysis, where computational hurdles and complex data structures are often faced.

Xu and Chen [73] showed that, with appropriate u^{-1} and $\boldsymbol{\beta}^{(0)}$, the IHT updates lead to a local maximum of problem (2.1), which provides an index set of k important features, $\widehat{s}$, corresponding to the non-zero entries of $\boldsymbol{\beta}^{(t)}$. Based on $\widehat{s}$, we then obtain a refined feature set from $\boldsymbol{X}$ for subsequent in-depth model fitting. Under some regularity conditions, $\widehat{s}$ contains all the relevant features with probability tending to one even when $p \gg n$, and thus is consistent for feature screening (sure screening).

The IHT updates can be viewed as a member of the Majorize-Minimization algorithms. Its practical performance is affected by a series of implementation decisions such as the choice of initial value, stopping criterion, screening size k, and u-search. We address those algorithm details in Section 2.3.2.

Algorithm 2.2 Post-screening Selection with SMLE

Input: Data $(\boldsymbol{y}, \boldsymbol{X}_s)$, selection criterion, sparsity lower bound k_{min}, sparsity upper bound k_{max}.

1. **For** k from k_{min} to k_{max}:
   (a) Obtain a sub-model s_k by running Algorithm 1 with sparsity k on $(\boldsymbol{y}, \boldsymbol{X}_s)$.
   (b) Evaluate s_k by computing a score C_k based on the input selection criterion.

Output: Sub-model $s^* \in \{s_{k_{min}}, \ldots, s_{k_{max}}\}$ with the smallest evaluation score C_k.

2.2.3 Post-screening selection with SMLE

In Xu and Chen [73], SMLE is mainly proposed for feature screening, the goal of which is to remove most irrelevant features before a more elaborate analysis. In practice, it is very likely that the feature set retained after screening still contains some irrelevant features. In principle, users can apply any well-developed selection method on the retained feature set to further identify relevant features.

In particular, one may further use the idea of SMLE to conduct post-screening selection. Specifically, assume that SMLE-screening was done and q features were retained; we obtain a refined $n \times q$ feature matrix $\boldsymbol{X}_s$. When q is moderate, we can conveniently obtain a series of sub-models by running Algorithm 1 on $(\boldsymbol{y}, \boldsymbol{X}_s)$ with sparsity k varying from $k_{min} \geq 1$ to $k_{max} \leq q$. A final sub-model can then be selected based an information criterion such as AIC [1], BIC [54], and EBIC [7]. We summarize this post-screening selection method in Algorithm 2, which inherits all the numerical merits from Algorithm 2.1. In particular, the joint information in $\boldsymbol{X}_s$ is naturally accounted for in the selection process.

Technically, Algorithm 2.2 can be used directly on $\boldsymbol{X}$ without the need for feature

screening. Its performance is actually quite impressive in our simulation studies. Nevertheless, when p is very large, we do recommend using Algorithm 2.2 only after Algorithm 2.1 for improved accuracy and stability.

2.3 Implementation details

The R package **SMLE** provides a set of functions for ultrahigh-dimensional feature screening under generalized linear models. **SMLE** can be installed from the Comprehensive R Archive Network (CRAN) at https://CRAN.R-project.org/package=SMLE using the following commands:

```
install.packages("SMLE")
library("SMLE")
```

SMLE requires the R packages **glmnet** [19] for parameter initialization, **mvnfast** [18] and **matrixcalc** [40] for simulating correlated data.

In this section, we first briefly describe the anticipated work flow for a data analysis using **SMLE**. We then provide a brief overview of the main functions of **SMLE** and illustrate the use and output of the functions through simple function calls. Detailed numerical examples are provided in Section 2.4.

2.3.1 SMLE work flow

The main **SMLE** functions are listed in Table 2.1. The work flow for a typical data analysis with **SMLE** is illustrated in Figure 2.1 and is summarized as follows:

Function name	Description
`Gen_Data()`	Simulate ultra-high dimensional GLM data
`SMLE()`	Joint feature screening using the SMLE method
`smle_select()`	Post-screening feature selection
`predict()`	Fitted or predicted values under the final model
`plot()`	Plot method to evaluate SMLE screening/selection for objects of class '`smle`' or '`selection`'

Table 2.1: Main **SMLE** functions and their brief descriptions.

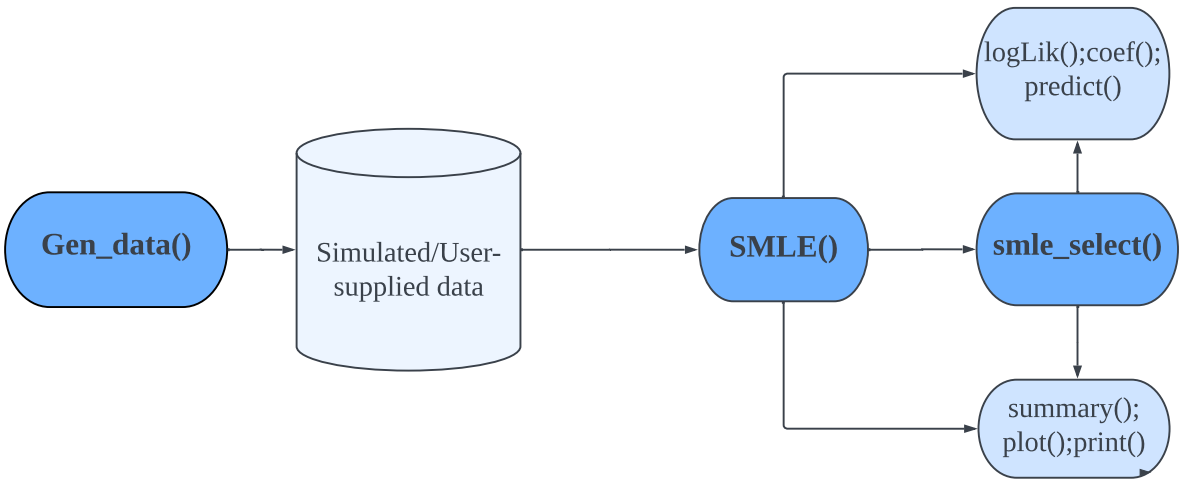


Figure 2.1: Flowchart for calling functions in the **SMLE** package.

For simulation studies or testing purposes, the function `Gen_Data()` can be used to simulate data under the assumption that the response variable follows a generalized linear model (GLM) that depends on a small subset of possibly correlated features. The function returns an object of class '`sdata`', which contains the matrix of features, the response variable, and additional information about the model used to generate the data.

The main function of the package, `SMLE()`, implements Algorithm 2.1 for screening out features that are unlikely to be related to the response variable. Users can pass a '`sdata`'

object, a data frame, or a matrix containing the data to the function. `SMLE()` returns an '`smle`' object that includes the top k features and additional information about the screening process.

For accurate post-screening feature selection, the function `smle_select()`, based on Algorithm 2.2, can be employed. Users can apply a selection criterion such as AIC, BIC, or EBIC. This function accepts either a '`smle`' object or a data frame (matrix). The result is an object of class '`selection`', providing details about the selected features and the selection process. Additionally, `smle_select()` can be run as a built-in option within `SMLE()` by setting the argument `selection = TRUE`.

The `plot()` function can be used to visualize the screening or selection results for both '`smle`' and '`selection`' objects.

The `predict()` function returns the fitted or predicted response values based on the features retained in either a '`smle`' or '`selection`' object.

Finally, the package includes several other functions similar to existing R functions for summarizing objects and fitting regression models, such as `summary()`, `coef()`, and `logLik()`.

2.3.2 Main functions and arguments

2.3.2.1 Simulating ultrahigh-dimensional GLM data

The motivation for the function `Gen_Data()` is to provide a freely available tool for simulating ultrahigh-dimensional GLM datasets with complex correlation structures between

features. Some of the correlation structures available in this function were used in Xu [73] to compare feature screening approaches. Both numerical and categorical features are permitted, and the number of features can be larger than the sample size. Users can choose the sample size n and the total number of features p in the dataset.

Numerical features are sampled from a normal distribution with one of four commonly-used correlation structures. The strength of correlation is controlled by a parameter ρ, which can optionally be set by the user using the argument `correlation`. The four different correlation structures are:

Independence (ID) All features are independently sampled from a standard normal distribution.

Moving average (MA) Features are jointly normal with covariance (correlation) ρ between adjacent features, $\rho/2$ between features two indices apart, and 0 otherwise. For example, the covariance matrix for four features would be:

$$\begin{bmatrix} 1 & \rho & \rho/2 & 0 \\ \rho & 1 & \rho & \rho/2 \\ \rho/2 & \rho & 1 & \rho \\ 0 & \rho/2 & \rho & 1 \end{bmatrix}$$

Compound symmetry (CS) Features are jointly normal with covariance $\rho/2$ if both features are causally related (relevant) to the response variable, and with covariance ρ otherwise. For example, with four features and assuming that features one and

four are relevant, the covariance matrix used to simulate the features would be

$$\begin{bmatrix} 1 & \rho & \rho & \rho/2 \\ \rho & 1 & \rho & \rho \\ \rho & \rho & 1 & \rho \\ \rho/2 & \rho & \rho & 1 \end{bmatrix}$$

Auto-regressive (AR) Features are jointly normal with covariance $\text{cov}(x_j, x_h) = \rho^{|j-h|}$ for the jth and hth features with $j, h \in \{1, \ldots, p\}$.

Categorical features are generated by first binning a numerical feature that was simulated as described above. The number of groups (levels) for a categorical feature is specified with `level_ctgidx`. After binning, the feature is converted from class '`numeric`' to '`factor`' and each bin is assigned a character from 'A' to 'Z'. Users are able to specify the number of categorical features in the dataset with the argument `num_ctgidx`, and their positions in the dataset with the argument `pos_ctgidx`.

The response variable is simulated by assuming a GLM and that only a subset of the features are influential on the response. Normal, binary, and Poisson response variables are all available; the model is chosen with the argument `family`, as with the R function `glm()`. The user can choose the number of influential features with the argument `num_truecoef`, and optionally, their positions in the feature matrix and their model effects with the arguments `pos_truecoef` and `effect_truecoef`, respectively. If the positions of the influential features are not provided, they would be chosen randomly.

`Gen_Data()` returns an object of class '`sdata`' containing the response vector $\boldsymbol{y}$, the

$n \times p$ feature matrix $\boldsymbol{X}$, and the coefficients for the features affecting the response.

The following code shows how to simulate a dataset with $n = 200$ observations and $p = 1000$ features, the first three of which are categorical, with three, four, and five levels, respectively. The response variable is generated based on a normal linear model with five influential features chosen by the function default.

```
R> set.seed(1)
R> Data_ctg <- Gen_Data(n = 200, p = 1000, family =
"gaussian", pos_ctgidx = c(1, 2, 3), level_ctgidx = c(3, 4, 5))
R> head(Data_ctg$X)[, 1:5]
```

```
  C1 C2 C3          X4         X5
1  A  B  C -1.29171904  0.1370216
2  A  B  D  0.90967045 -1.2996452
3  B  B  A -1.10775568  1.1514089
4  B  C  C -0.38412387  1.5134475
5  B  C  D  0.08273483  0.8021379
6  B  A  D -0.48388247 -0.4305369
```

2.3.2.2 Joint feature screening

The main goal is to identify a manageable set of $k < p$ features that are most related to the response variable. To that end, `SMLE()` is used to screen out features unlikely to be

influential (i.e., irrelevant features); it serves as a pre-processing step before an in-depth analysis. Users can pass information about the input data $(\boldsymbol{y}, \boldsymbol{X})$ to `SMLE()` via a '`sdata`' object, a data frame, or data matrices. When data are not input from a '`sdata`' object, the user should further specify the type of GLM with the argument `family`. The function conducts effective feature screening based on Algorithm 2.1, which naturally incorporates the joint effects among features. In `SMLE()`, we make use of the R function `crossprod()` to handle the ultrahigh dimensional matrix product involved in Algorithm 2.1 without doing matrix transpose; this leads to improved computational efficiency in comparison with the original implementation in [73]. In Figure 2.2, we show the averaged elapsed time (in seconds) of `SMLE()` and the original IHT code on a series of datasets generated by `Gen_Data(correlation = "ID")` with $n = 200$ and p varying from 4000 to 20000. The running time is based on 100 repetitions.

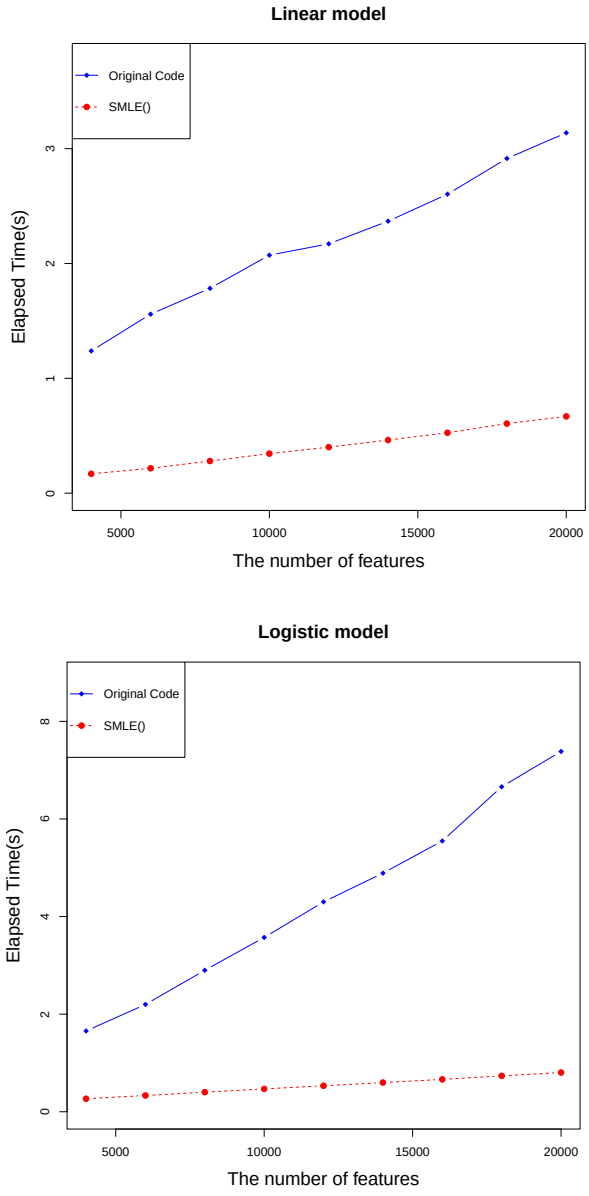


Figure 2.2: Running time comparison between `SMLE()` and the original IHT implementation in Xu [73].

In Table 2.2, we list the main arguments of `SMLE()` for specialized users to control the

screening process. In particular, the argument `k` controls the number of important features to be retained from $\boldsymbol{X}$ after screening. The choice of `k` should reflect a user's belief or prior knowledge on the total number of relevant features for the input data. Intuitively, a larger `k` increases the chance of retaining all relevant features, while a smaller `k` brings more interpretive value and computational convenience for the subsequent in-depth analysis. One practical strategy is to set `k` to be three times larger than the anticipated number of relevant features. In `SMLE()`, the default value of `k` is the largest integer not exceeding $0.5\log(n)n^{1/3}$, which is recommended in [73] from a theoretical perspective.

The argument `coef_initial` allows users to specify an initial value $\boldsymbol{\beta}^{(0)}$ for SMLE-screening. Since Algorithm 2.1 may lead to a local maximum, an informative $\boldsymbol{\beta}^{(0)}$ helps to improve screening accuracy. This argument can be helpful in applications, where prior knowledge of the model coefficients are available. When the number of non-zero components of $\boldsymbol{\beta}^{(0)}$ is larger than p, `SMLE()` will truncate it to k, ensuring the non-decreasing property of the IHT algorithm. The default value of `coef_initial` is the Lasso [58] estimate with the largest sparsity not exceeding $n-1$, which is implemented by the function `glmnet(pmax = n-1)` in the R package **glmnet**. This data-driven choice is recommended in Xu and Chen [73] and is also supported by our numerical studies.

When a non-informative initialization is preferred, one may simply set $\boldsymbol{\beta}^{(0)} = 0$, which runs Algorithm 2.1 with a null model. In this case, `SMLE()` starts from scratch and iteratively detects important features via $\boldsymbol{\beta}^{(t)}$ during the IHT process (see Figure 2.4). Since a non-decreasing likelihood is obtained by IHT updates, technically $\boldsymbol{\beta}^{(t)}$ from Algorithm 2.1 can always be viewed as an improvement over $\boldsymbol{\beta}^{(0)}$ in terms of the joint likelihood. In particular, when SMLE is used on linear models with $\boldsymbol{\beta}^{(0)} = 0$, the screening result based on

$\boldsymbol{\beta}^{(1)}$ coincides with the SIS-screening based on marginal effects; $\boldsymbol{\beta}^{(2)}$ further improves SIS by incorporating joint information in $\boldsymbol{X}$. Readers may refer to Section 2.4.3 for a numerical comparison between the informative and non-informative initialization strategies.

In `SMLE()`, the initial value of u^{-1} is determined as follows. When $\boldsymbol{\beta}^{(0)} = 0$, we simply set $u^{-1} = U/\sqrt{p}$ with $U > 0$ being a magnification parameter. When $\boldsymbol{\beta}^{(0)} \neq 0$, we generate a sub-matrix $\boldsymbol{X}_0$ using the columns of $\boldsymbol{X}$ indicated by $\{j : \beta_j^{(0)} \neq 0\}$ and set

$$u^{-1} = \frac{U}{\sqrt{p}\|\boldsymbol{X}_0\|_\infty^2},$$

where $\|.\|_\infty$ is the matrix infinity norm denoting the maximum absolute row sum of a matrix. The use of $\sqrt{p}\|\boldsymbol{X}_0\|_\infty^2$ serves as a computationally convenient replacement for computing the largest eigenvalue of $\boldsymbol{X}^T\boldsymbol{X}$, which is used in Xu [73] as a theoretical guidance for choosing u. Users can specify the magnification parameter U and the decrease rate $\tau \in (0, 1)$ in u-search with the arguments `U` and `U_rate`, respectively. The default values are `U = 1` and `U_rate = 0.5`.

`SMLE()` terminates the IHT iterations when $\|\boldsymbol{\beta}^{(t)} - \boldsymbol{\beta}^{(t-1)}\|_2$ is below the tolerance level specified in the argument `tol`. Since our goal here is feature screening, a large number of iterations t may not always be necessary. We observe that, in many of our numerical examples, `SMLE()` can successfully identify all the relevant features within a few iterations by estimating the corresponding coefficients to be non-zero. Therefore, when an accurate coefficient estimate is not needed, users may choose to set the argument `fast = TRUE`, which allows early stopping of Algorithm 2.1. Specifically, with `fast = TRUE`, `SMLE()` terminates the IHT iterations when one of the following rules is satisfied:

Arguments	Description	Default value
`k`	Total number of features to be retained after screening.	$\frac{1}{2}\log(n)n^{1/3}$
`keyset`	A vector to indicate a set of key features that are forced to remain in the model.	NULL
`coef_initial`	Initial coefficient value $\boldsymbol{\beta}^{(0)}$ for IHT.	`glmnet(pmax=n-1)`
`categorical`	Logical flag for whether the input feature matrix includes categorical features.	NULL
`group`	Logical flag for whether to treat the dummy variables corresponding to each categorical feature as a group.	TRUE
`tol`	A tolerance level for $\|\boldsymbol{\beta}^{(t)} - \boldsymbol{\beta}^{(t-1)}\|_2$ to stop the iteration.	10^{-2}
`fast`	Logical flag to enable early stop for IHT.	FALSE
`U_rate`	Decrease rate in u-search.	0.5
`U`	Initial magnification level in u-search.	1

Table 2.2: Main arguments for the `SMLE()` function.

- $\|\boldsymbol{\beta}^{(t)} - \boldsymbol{\beta}^{(t-1)}\|_2 < \mathtt{k}^{1/2} \times \mathtt{tol}$,
- $l(\boldsymbol{\beta}^{(t)}) - l(\boldsymbol{\beta}^{(t-1)}) < 0.01\ [\ l(\boldsymbol{\beta}^{(2)}) - l(\boldsymbol{\beta}^{(1)})\ ]$,
- The non-zero entries in $\{\boldsymbol{\beta}^{(t)}\}$ remain unchanged for 10 consecutive iterations.

When the input data for `SMLE()` contains categorical features, users should set the argument `categorical = TRUE`, which enables the function to automatically detect the locations and levels of the `factor` columns in the feature matrix. By default, a L-level categorical feature would be encoded by $L-1$ dichotomous dummy variables, which are to be retained or screened out together if `group = TRUE`.

With the `keyset` argument, users can choose to manually keep a subset of features in the SMLE-screening process; this can be useful when some features are known to be influential or confounding. The following code demonstrates the usage of `SMLE()` with

`keyset` on the dataset `Data_ctg` generated in Section 2.3.2. Here we ensure that the first (`C1`), the fourth (`X4`), and the fifth (`X5`), features are always kept during the IHT updates and are therefore surely retained after screening. In this example, since we choose to retain `k = 15` features in total, the retained set would include the 3 features specified in `keyset` plus 12 features to be suggested by Algorithm 2.1. When `keyset` contains categorical features, it is required to have `group = TRUE`.

```
R> fit  <-  SMLE(Y = Data_ctg$Y, X = Data_ctg$X, k = 15, family = "gaussian",
keyset = c(1, 4, 5), categorical = TRUE, group = TRUE)
```

`SMLE()` returns an object of class '`smle`', which contains information regarding the input data, model assumption, IHT updates, and the screening results.

2.3.2.3 Post-screening selection

As mentioned in Section 2.2.3, the features retained after screening are still likely to contain some that are not related to the response. The function `smle_select()` is designed to implement Algorithm 2.2 to further identify the relevant features.

`smle_select()` can be applied to a '`sdata`' object, a '`smle`' object, or a user-supplied dataset $(\boldsymbol{y}, \boldsymbol{X}_s)$. As discussed before, when p is very large, a screening step is usually needed before `smle_select()` can be efficiently applied. In the package, we provide an option to automatically run `smle_select()` after screening within the main function `SMLE()`.

We list the main arguments of `smle_select()` in Table 2.3. Users can set the range or specify a subset of features to be further selected. With the argument `criterion`, users can choose to run `smle_select()` using their preferred selection criterion.

Arguments	Description	Default value
k_min (k_max)	The lower (upper) bound for candidate model sparsity.	k_min = 1, k_max = number of input features
sub_model	A index vector indicating which features are to be selected. Not applicable if a 'smle' object is the input.	NULL
criterion	Selection criterion to be used. One of "ebic","bic","aic".	"ebic"
gamma_ebic	The EBIC parameter in $[0, 1]$.	0.5
gamma_seq	The sequence of values for gamma_ebic when vote = TRUE.	(0,0.2,0.4,0.6,0.8,1)
vote	The logical flag for whether to perform the voting procedure, when criterion = "ebic".	FALSE
vote_threshold	A relative voting threshold in percentage. A feature is considered to be important when it receives votes passing the threshold.	0.6
para	Logical flag to use parallel computing to do selection.	FALSE

Table 2.3: Main arguments for the smle_select() function.

When the criterion EBIC is used, users may further specify its tuning parameter using the argument `gamma_ebic`. Alternatively, one may set `vote = TRUE`, which would repeat the EBIC-based selection with a sequence of tuning parameters provided in `gamma_seq`. With different tuning parameters, different sets of features would be selected; a feature is considered to be important when it is selected more than `vote_threshold` of times in the entire procedure. Users may change this threshold to enlarge or reduce the size of the selected model by using the `vote_update()` function, which is a simple method for a '`selection`' object that avoids the need to re-run `smle_select()`.

The following code demonstrates the use of `smle_select()` with EBIC voting on the '`smle`' object `fit` generated before. The output `fit_s` is an object of class '`selection`', which contains the information regarding the selection procedure and result.

```
R> fit_s <- smle_select(fit, criterion = "ebic", gamma_seq = seq(0, 1, 0.2),
vote = TRUE)
```

In `smle_select()`, the selection is done by evaluating a sequence of sub-models with sizes varying from `k_min` to `k_max`. The arguments `k_min` and `k_max` allow the users to control the searching range based on their prior knowledge or expectation about the model size.

When the package is run in a multi-core computing environment, users may opt to use parallel computing to boost this selection procedure. Specifically, by setting `para = TRUE`, the creation and evaluation of the sub-models would be distributed to multiple computing cores for parallel processing. The implementation of the parallel option uses the **parallel** R package [50] and depends on the users operating system: for Unix and Mac users

`mclapply()` is used and for Windows `parLapply()` is used. Note that `smle_select()` is mainly designed for post-screening selection, where the number of candidate features is moderate and the associated computational cost is often acceptable. Considering the communication cost between cores, parallel processing may not necessarily lead to a significant speed-up for the regular usage of `smle_select()`.

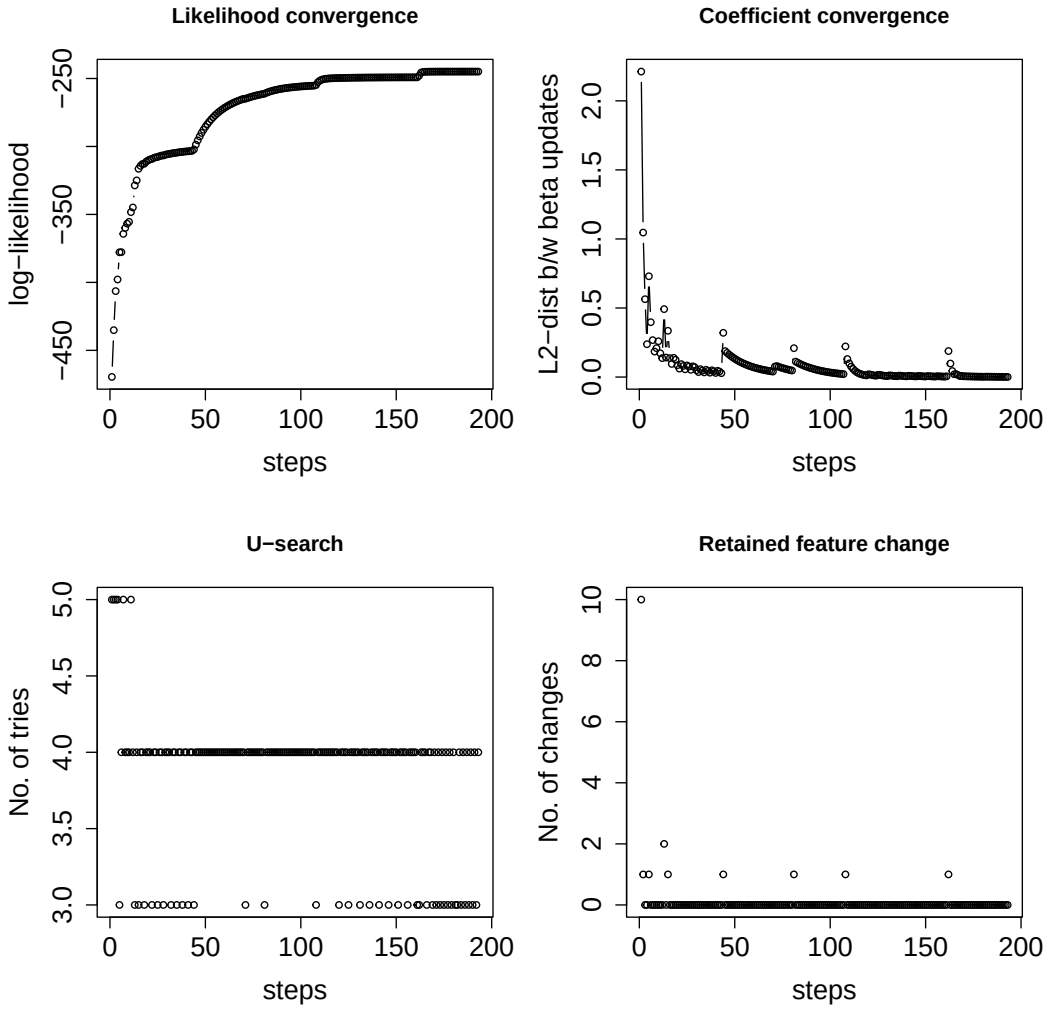


Figure 2.3: Example plot for '`smle`' class - IHT Convergence with $\boldsymbol{\beta}^{(0)} = 0$.

2.3.2.4 Plotting

Plot functions have been included for both the '`smle`' and '`selection`' classes. The `plot()` function for class '`smle`' returns two plot windows. By default, the first plot window contains four subplots each showing a measure of IHT convergence on the vertical axis and iteration index on the horizontal axis; see Figure 2.3 as an example. The convergence measures are:

- log-likelihood (top left),
- Euclidean distance between the current and previous coefficient estimates (top right),
- the number of u-search tries in Algorithm 2.1 (bottom left),
- the number of membership changes in the retained feature set between the current and the previous IHT updates (bottom right).

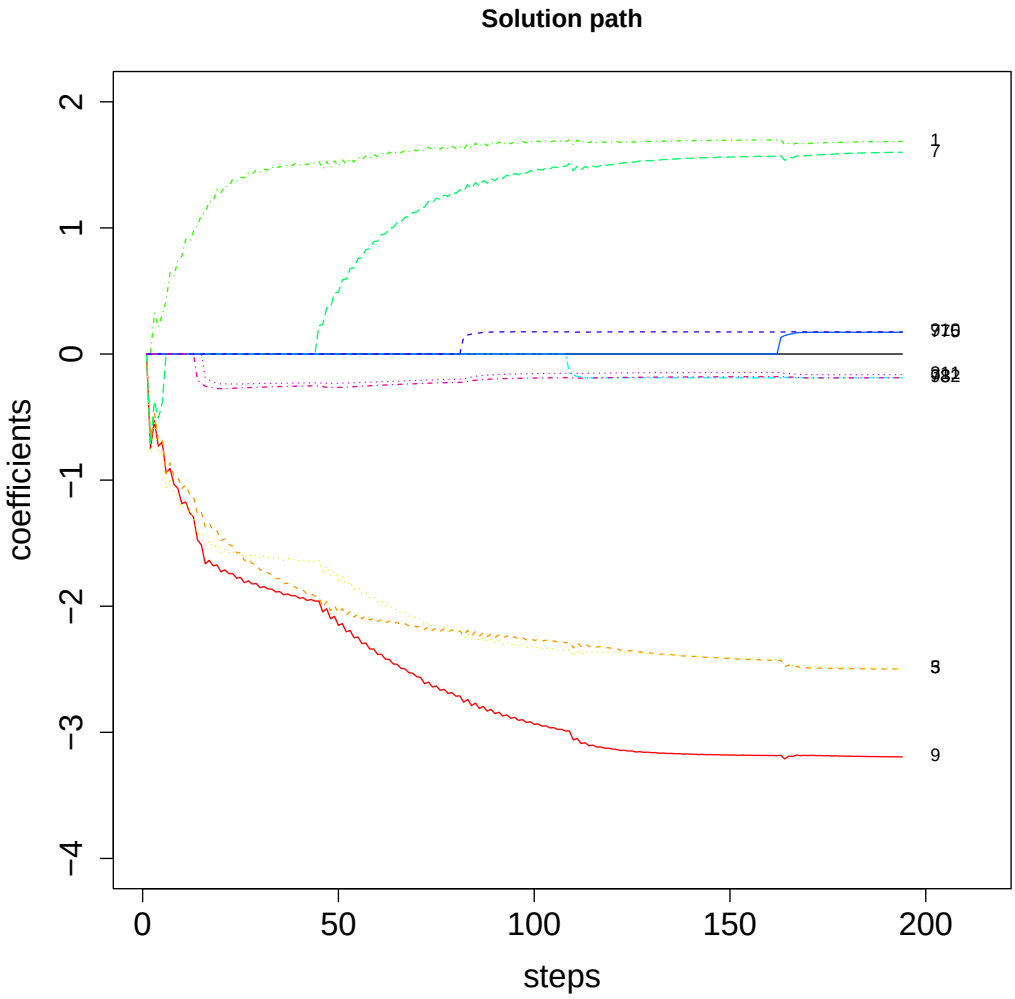
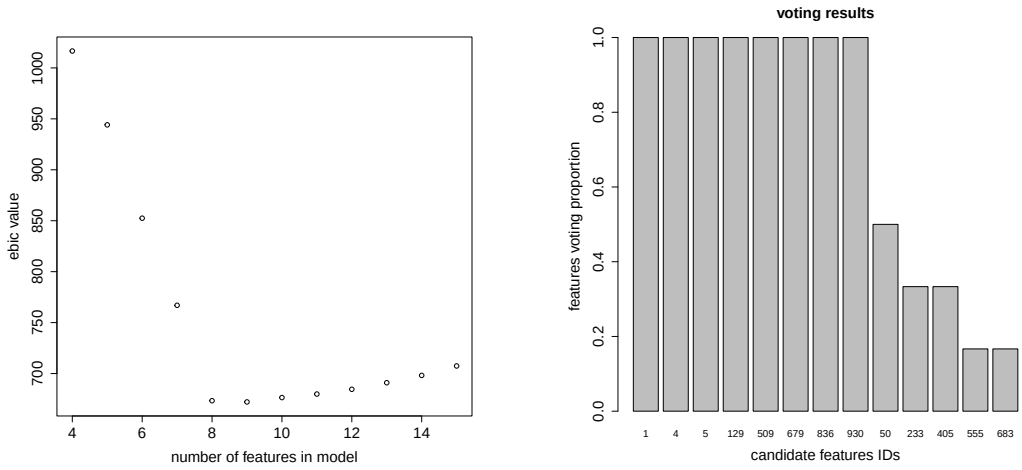


Figure 2.4: Example plot for '`smle`' class - IHT solution path with $\boldsymbol{\beta}^{(0)} = 0$.

The second plot window shows the solution path (estimated coefficient by IHT iteration) for selected features; see Figure 2.4 as an example. This plot provides a direct insight on how a coefficient changes over the iterations. By default, the solution path is given for all the relevant features suggested in the input '`smle`' object. Users may choose to plot a

solution path for a customized group of features. This can be done by plotting only the top `num_path` features or plotting only the features specified in the argument `which_path`.

Users can optionally change which figure appears in the second plot window. For example, passing the argument `outplot = 1` will cause the log-likelihood to appear in the second plot window; the solution path will instead be plotted in the first plot window. Any additional arguments passed to `plot()` will be used when creating the plot in the second plot window.



Selection criterion Voting results

Figure 2.5: Example plots for the `selection` class, illustrating the selection criterion and voting results.

The `plot()` function for an object of '`selection`' class returns a plot showing the selection criterion scores evaluated for the candidate sub-models with varying sizes (number of features); see Figure 2.5(a) as an example. A selection curve typically has a "V" shape: the curve decreases at the beginning as sub-model size increases before it increases again when there is little benefit in including additional features. For example, Figure 2.5(a)

suggests a sub-model with six features, which are to be treated as influential ones. When criterion EBIC is used with `vote = TRUE`, `plot()` additionally returns a voting plot showing the relative inclusion frequency of each feature in the sequence of sub-models generated by Algorithm 2.2; see Figure 2.5(b).

2.3.2.5 Prediction

The `predict()` function in **SMLE** is based on the generic `predict()` function in **stats**; it returns predicted response values based on a model containing only the features retained in a '`smle`' object or selected in a '`selection`' object. The predictions are made by fitting the model using the `glm()` function.

As with the `predict()` function for class '`glm`' or '`lm`', users can optionally specify a data frame with new values for the features in the model using the `newdata` argument. If `newdata` is not provided, predictions will be given for the original training data; this leads to the fitted response values.

When using `predict()`, the type of prediction required must be specified using the `type` argument. The default is `type = link` which returns predictions on the scale of the linear predictor; `type = response` returns predictions on the scale of the response variable. In particular, with `type = response`, `predict()` returns the predicted mean responses for a linear or Poisson model and the predicted success probabilities for a logistic model.

2.3.2.6 Additional functions for regression-based objects

SMLE has some additional functions that can be used to extract information about the

model fit. Specifically, the function `logLik()` first refits the screened (or selected) model using `glm()` and then returns the log-likelihood of the fitted model. The function `coef()` can be conveniently used to extract regression coefficients from a fitted '`smle`' or '`selection`' object. In addition, `summary()` is extended from the base function to take a '`smle`' or '`selection`' object and displays summary information about the fitted model object.

2.4 Examples

We demonstrate the usage and effectiveness of **SMLE** using a series of simulation studies. Numerical experiments were conducted on a Mac laptop with M1 chip and 8 GB memory and were performed using R Statistical Software (v4.2.2; [50]).

2.4.1 Demo code for SMLE-screening

We first show how to use **SMLE** to conduct feature screening and post-screening selection via a simulated example. To this end, we use `Gen_Data()` to generate a synthetic dataset with $n = 400$ observations and $p = 1000$ features. We generate the feature matrix $\boldsymbol{X}$ from a multivariate normal distribution with an auto-regressive structure, where the adjacent features have a high correlation of $\rho = 0.9$.

$$Y = 4x_1 + -5x_3 + 3x_5 + 4x_7 - 5x_9.$$

To achieve a more intricate structure, we merge 5 datasets. We do this by summing up their responses to form the Y variable, and by combining all their features to create the X

matrix.

In this setup, the feature matrix contains only five features that are causally-related to the response, as indicated in the model.

This dataset is generated by the following code; the resulting simulated data are stored in `Data_eg`, which is an object of class '`sdata`'. Users can use the `print()` function to show properties of the simulated data.

```
R> set.seed(1)
R> Data_eg <- Gen_Data(n = 400, p = 1000, family = "binomial",
correlation = "AR", rho = 0.9, pos_truecoef = c(1, 3, 5, 7, 9),
effect_truecoef = c(2, 3, -3, 3, -4))
R> print(Data_eg)

Call:
 Gen_Data(n = 400, p = 1000, pos_truecoef = c(1, 3, 5, 7, 9),
    effect_truecoef = c(2, 3, -3, 3, -4), correlation = "AR",
    rho = 0.9, family = "binomial")


An object of class sdata


Simulated Dataset Properties:
 Length of response: 400
 Dim of features: 400 x 1000
 Correlation: auto regressive
```

```
 Rho: 0.9
 Index of Causal Features: 1, 3, 5, 7, 9
 Model Type: binomial
```

We then run `SMLE()` to conduct SMLE-screening on the data example `Data_eg`, assuming that the identities of the causal features were unknown; the goal is to obtain a refined feature set by removing most irrelevant features from $\boldsymbol{X}$. The following code shows the simplest function call to `SMLE()`, where we aim to retain only $k = 10$ important features out of $p = 1000$.

```
R> fit1 <- SMLE(Y = Data_eg$Y, X = Data_eg$X, k = 10, family = "binomial",
coef_initial = rep(0, 1000))
R> summary(fit1)


Call:
 SMLE(X = Data_eg$X, Y = Data_eg$Y, k = 10, family = "binomial",
 coef_initial = rep(0, 1000))

An object of class summary.smle

Summary:

  Length of response: 400
```

```
Dim of features: 400 x 1000
Model type: binomial
Model size: 10
Feature name: 1, 3, 5, 7, 9, 10, 404, 470, 535, 661
Feature index: 1, 3, 5, 7, 9, 10, 404, 470, 535, 661
Intercept: 0.2793397
Coefficients estimated by IHT: 1.894, 3.534, -2.648, 3.517, -4.617, -0.615,
-0.429, -0.549, 0.382, 0.578
Number of IHT iteration steps: 252
```

The function returns a '`smle`' object in the variable called `fit1`. The `summary()` function confirms that a refined set of 10 features is retained after 252 IHT iterations. We can see that all 5 causal features used to generate the response are retained in the refined set. This indicates that screening is successful; the dimensionality of the feature space is reduced from $p = 1000$ down to $k = 10$ without losing any important information.

In the code that follows, we fit a marginal regression between the response and the second feature, x_2, in `Data_eg`. From the TRUE model, we know that x_2 is not causally-related to the response. Yet, we can see that the marginal effect of x_2 appears to be pretty high; thus, this irrelevant feature is likely to be retained in the model if the screening is done based on marginal effects only. In this example, `SMLE()` accurately removes x_2, as its screening naturally incorporates the joint effects among features.

```
R> with(Data_eg, coef(summary(glm(Y ~ X[,2], family = "binomial"))))

              Estimate Std. Error   z value     Pr(>|z|)
(Intercept) 0.02440072  0.1125979 0.2167067 8.284369e-01
X[, 2]      1.10465766  0.1337063 8.2618250 1.434619e-16
```

Note that the refined set returned in `fit1` still contains some irrelevant features; this is to be expected, as the goal of feature screening is merely to remove most irrelevant features before conducting an in-depth analysis. As discussed in Section 2.2.3, one may conduct an elaborate selection on the refined set to further identify the causal features. In particular, the `smle_select()` function in **SMLE** can be readily used for the purpose of post-screening selection. The following code shows how this function can be used on `fit1` with the selection criterion EBIC. As can be seen below, `smle_select()` returns a '`selection`' object `fit1_s`, which exactly identifies the five features in the TRUE data generating model.

```
R> fit1_s <- smle_select(fit1, criterion = "ebic")
R> summary(fit1_s)

Call:
  smle_select(object = fit1, criterion = "ebic")

An object of class summary.selection

Summary:
```

```
Length of response: 400
Dim of features: 400 x 1000
Model type: binomial
Selected model size: 5
Selected features: 1, 3, 5, 7, 9
Selection criterion: ebic
Gamma for ebic: 0.5
```

As mentioned in Section 2.3.2, users can visualize the above screening and selection results by using the `plot()` function. Users can also make use of the arguments such as `keyset` or `k_min` to refine the above results.

2.4.2 Screening performance

Next, we assess the performance of **SMLE** in terms of screening accuracy and efficiency. To this end, we use `Gen_data()` to generate 500 independent datasets with $p = 2000$ and $n = 100, 250, 600$ from linear, Poisson, and logistic models, respectively. For all these datasets, a compound symmetry structure with $\rho = 0.3$ is used, where the first four features (i.e., x_1, x_2, x_3, x_4) are used to generate the response variable. An equal coefficient is assigned to each of the four causal features; the value of the coefficient is set to 2.5, 0.7, and 1.1 for the three aforementioned models, respectively.

We use `SMLE()` in our package to conduct feature screening on these simulated datasets to retain only $k = 20, 10, 30$ important features for linear, Poisson, and logistic models,

respectively. We measure the screening accuracy by sure screening rate (SSR) and positive retaining rate (PRR). SSR is reported as the proportion of times that all causal features are retained after screening; PRR is calculated by the averaged proportion of causal features that are retained after screening. An averaged elapsed time (in seconds) for `SMLE()` in each model setup is reported as a measure of screening efficiency. For comparison, the performances of (I)SIS, DC-SIS, and Lasso are also reported under the same setup. Moreover, we repeat the experiments using the method of Abess, which was proposed in Zhu [87] for the high-dimensional best subset selection. Following the recommendations from the corresponding packages, we implement SIS by `SIS(iter = F)` and ISIS by `SIS(iter = T)` in package **SIS**, implement DC-SIS by `screenIID()` in package **VariableScreening**, implement Lasso by `glmnet()` in package **glmnet**, and implement Abess by `abess()` in package **abess** [86].

We summarize the simulation results in Table 2.4, where all decimal numbers are rounded to 2 digits. It can be seen that `SMLE()` has the highest accuracy for all three model setups in terms of both SSR and PRR; the other methods (packages) seem to be heavily affected by the correlation among the features. By the nature of Algorithm 2.1, the high accuracy of `SMLE()` comes with a computational cost, but this is moderate in most cases. Considering the improved accuracy of `SMLE()`, this small computational investment seems to be quite worthwhile.

Model	Functions	PRR	SSR	Time (s)
Linear	`SIS(iter = F)`	0.21	0.00	0.02
	`SIS(iter = T)`	0.73	0.60	1.36
	`screenIID()`	0.18	0.00	0.41
	`glmnet()`	0.37	0.04	< 0.01
	`abess()`	0.65	0.46	0.01
	`SMLE()`	1.00	1.00	0.01
Poisson	`SIS(iter = F)`	0.08	0.00	0.05
	`SIS(iter = T)`	0.51	0.38	3.26
	`screenIID()`	0.50	0.00	2.21
	`glmnet()`	0.09	0.00	0.01
	`abess()`	0.63	0.51	0.03
	`SMLE()`	0.93	0.85	0.81
Logistic	`SIS(iter = F)`	0.53	0.04	0.15
	`SIS(iter = T)`	0.65	0.37	9.09
	`screenIID()`	0.50	0.00	14.69
	`glmnet()`	0.78	0.43	0.02
	`abess()`	0.91	0.81	0.14
	`SMLE()`	0.94	0.82	0.10

Table 2.4: Comparison of screening performance for different models using `SIS()`, `glmnet()`, `screenIID()`, `abess()`, and `SMLE()`. PRR represents the Positive Rate Ratio, SSR is the Sensitivity-Specificity Ratio, and Time is measured in seconds.

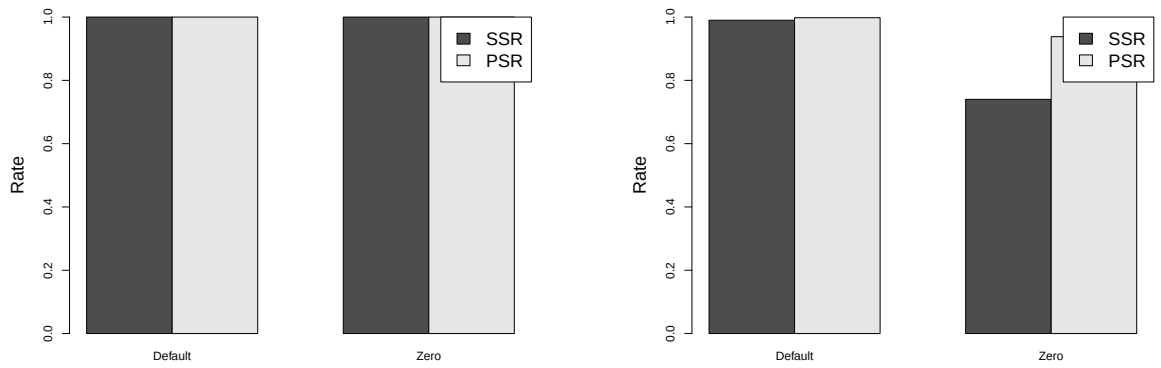
2.4.3 Impacts of initialization

As discussed in Section 2.3.2, a good choice of $\boldsymbol{\beta}^{(0)}$ may be beneficial for SMLE-screening. To provide some insights, we run `SMLE(k = 20)` on simulated datasets with `coef_initial` being the default value or a zero vector. The datasets in this example are generated by the following code.

```
R> Data_S1 <- Gen_Data(n = 300, p = 1000, family = "gaussian",
correlation = "ID", pos_truecoef = c(1, 3, 5, 7, 9),
effect_truecoef = c(2, 3, -3, 3, -4))

R> Data_S2 <- Gen_Data(n = 400, p = 1000, family = "gaussian",
correlation = "AR", rho = 0.9, pos_truecoef = c(1, 3, 5, 7, 9),
effect_truecoef = c(2, 3, -3, 3, -4))
```

Based on `Data_S1` and `Data_S2`, we calculate SSR and PSR of SMLE-screening with both default and zero start on 100 repetitions; the results are shown in Figure 2.6. It is seen that both initialization strategies lead to high screening accuracy on `Data_S1`, where features are independent with each other. The benefits of using an informative $\boldsymbol{\beta}^{(0)}$ is observed on `Data_S2`, where strong correlations present among the features. These findings seem to support the recommended value of `coef_initial` in our package; users may also make their own choice of `coef_initial` based on the prior knowledge about model parameters.



Data_S1 Data_S2

Figure 2.6: Screening accuracy of `SMLE()` with default and zero-vector start.

2.4.4 Application to high-dimensional genetic data

To further demonstrate **SMLE**, we applied it to a simulated high-dimensional genetic dataset to detect associations between single-nucleotide polymorphisms (SNPs) and a response variable. To create a realistic genetic dataset, the genotypes were sampled from genotypic distributions derived from the 1000 Genomes project, Phase 1 [57], using the R package **sim1000G** [8]. Since the SNP distributions are derived from individuals in existing human populations, this dataset naturally exhibits a complex correlation structure. Specifically, it is believed that strong non-linear correlations exist between SNPs in close proximity on the same chromosome.

The dataset consists of $p = 10,031$ SNPs from chromosomes 14 through 22 across $n = 800$ individuals. The genotypes were coded as 0, 1, or 2, representing the number of minor alleles (the allele that is less common in the sample). The continuous response

variable was simulated from a normal distribution, with the mean depending additively on the causal SNPs and a standard deviation of 8. The linear predictor for the response follows a polygenic model, specifying 31 SNPs with varying effect sizes. The goal is to identify regions containing these 31 causal SNPs, whose effects were determined as follows.

Twenty SNPs were randomly assigned small effects, which were drawn from a $N(0, 0.1)$ distribution. Five SNPs were randomly assigned moderate effects, drawn from a $N(0, 0.5)$ distribution. Four SNPs (rs2967291, rs1534941, rs112915930, rs6132052) were selected to have large effects, fixed at 2, -2, 1, and -1, respectively. Additionally, two SNPs (rs12608528 and rs11157211) were assigned interaction effects, with no marginal effect simulated; the interaction coefficient was set to 4. Finally, the intercept was set arbitrarily at 40 to avoid negative response values.

This simulated genetic dataset is included in **SMLE** and can be loaded with the following command.

```
R> data("synSNP")
```

We save the response as a vector `Y_SNP` and the features as a matrix `X_SNP`.

```
R> Y_SNP <- synSNP[, 1]
R> X_SNP <- as.matrix(synSNP[, -1])
```

The goal of study here is to flag genomic regions likely to contain the causal SNPs. This job can be typically done by identifying a small set of key SNPs and then marking the nearby locations. As a conventional approach, we first performed a single-SNP analysis using linear regression to test the marginal effects of the SNPs on the response. We

report $-\log_{10}$ of the p value of the corresponding estimated marginal coefficients using a Manhattan plot (Figure 2.7). The p value of the 31 causal SNPs are coloured based on their effect sizes. The blue line shows the conventional threshold used to declare suggestive significance (p value $< 10^{-5}$). We observed that only one SNP passed this threshold, which was simulated to have a large effect ("rs11157211"). All other SNPs did not show marginal significance and thus were not treated as the key ones. As a result, the conventional marginal-test-based approach failed to flag all the regions harboring the causal SNPs.

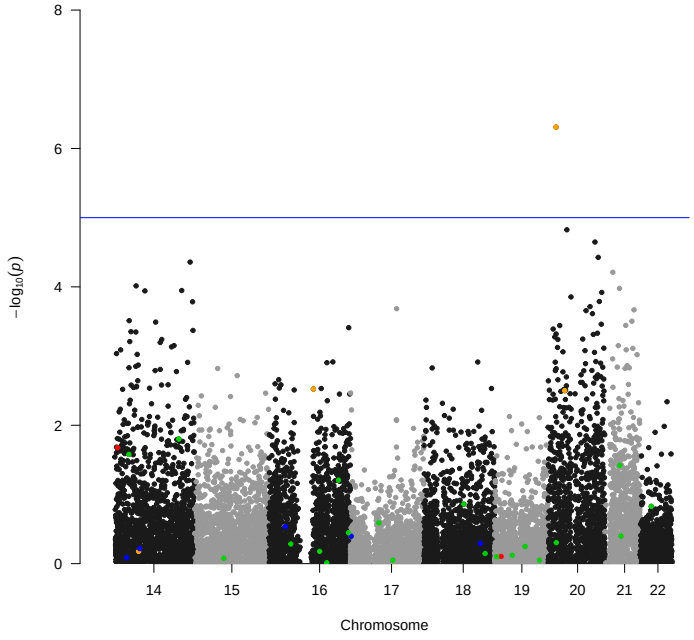


Figure 2.7: Manhattan plot of $-\log_{10}(p$ value) for the single-SNP association test by package **qqman** [59]. The blue line represents suggestive significance. The coloured points correspond to SNPs simulated to be causally related to the response in the order of importance: red, orange, blue, green.

We then ran `SMLE()` to flag genomic regions by retaining $k = 80$ key SNPs after

screening. The results were compared with those obtained by `SIS()` and `glmnet()`. The three methods were carried out with the following code.

```
R> SMLE_fit <- SMLE(Y = Y_SNP, X = X_SNP, family = "gaussian",
k = 80, fast = F)
```

```
R> SIS_fit <- SIS(y = Y_SNP, x = X_SNP, family = "gaussian",
nsis = 80, iter = F)
```

```
R> lasso_fit <- glmnet(x = X_SNP,y = Y_SNP, family = "gaussian",
pmax = 80 )
```

In Figure 2.8, we show the screening result from the code above. It is observed that `SIS()` focused on chromosomes 14, 20, and 21, where SNPs have relatively larger marginal effects; in contrast, `SMLE()` and `glmnet()` flagged regions across all chromosomes.

To assess the screening performance, for each of the 31 causal SNPs we compute the minimum distance between that SNP and the SNPs in the retained set (MRD). A small MRD indicates that the retained set contains at least one SNP that is close to the causal SNP; consequently, genetic regions suggested by the retained SNPs are likely to contain the causal SNPs. Therefore, an effective screening is expected to have small MRDs for most causal SNPs.

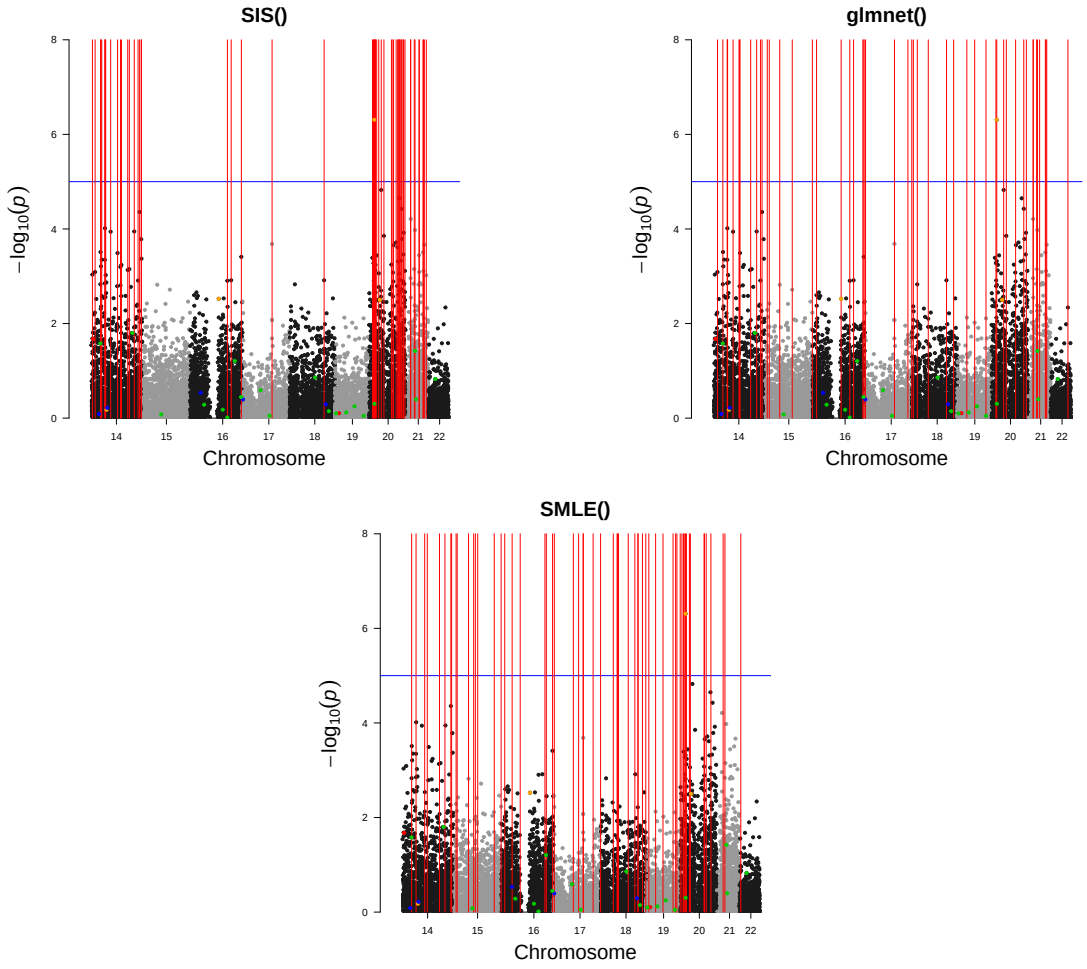


Figure 2.8: Important genetic regions flagged with red lines by `SIS()` (top left), `glmnet()` (top right), and `SMLE()` (bottom).

In Figure 2.9, we show the MRDs of the 31 causal SNPs from the screening conducted with `SMLE()`, `SIS()`, and `glmnet()`. The horizontal axis of the figure denotes the true effect sizes of the causal SNPs. Since SIS only uses the marginal information, the 80 locations suggested by `SIS()` tend to be very close to the peak on chromosome 20 as shown in

Figure 2.7; this leads to large MRDs (with a mean value of 176.1) for most causal SNPs that are away from this peak. Since `SMLE()` and `glmnet()` consider joint effects, they achieve much smaller MRDs, with the mean values given by 42.2 and 66.8 respectively. Compared with `glmnet()`, `SMLE()` leads to 33.8% improvement in the averaged MRD among all casual SNPs. The new package also shows an advantage in locating the two causal SNPs (rs12608528, rs11157211) with a large interaction effect.

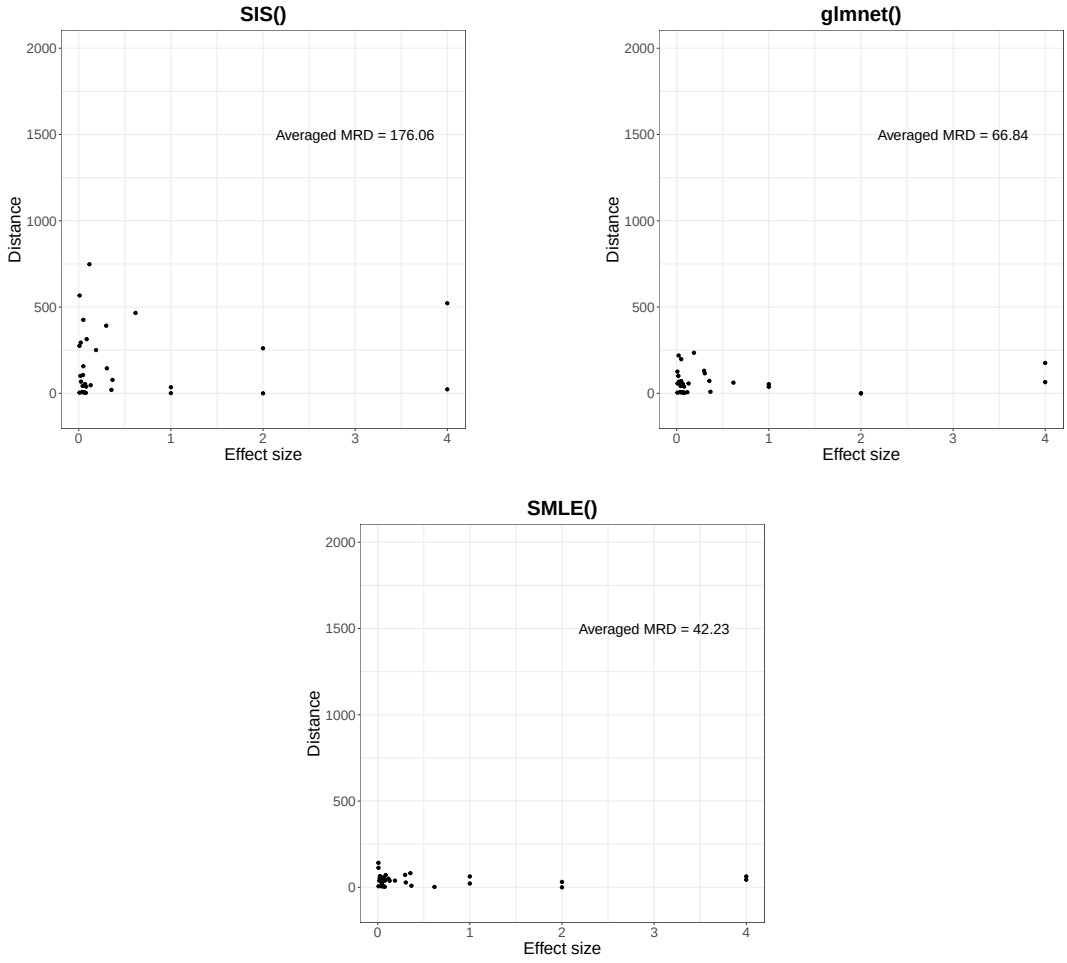


Figure 2.9: Averaged MRD on casual SNPs for `SIS()` (top left), `glmnet()` (top right), and `SMLE()`(bottom).

We repeated the analysis by increasing the model screening size, k. The corresponding averaged MRDs for all causal SNPs are shown in Figure 2.10. The performance of all three methods improve as k increases. Compared with the other two methods, `SMLE()` tends to be more stable; it conducts effective screening by consistently achieving a low

averaged MRD over different choices of k. The high reliability makes it a preferable choice in practice.

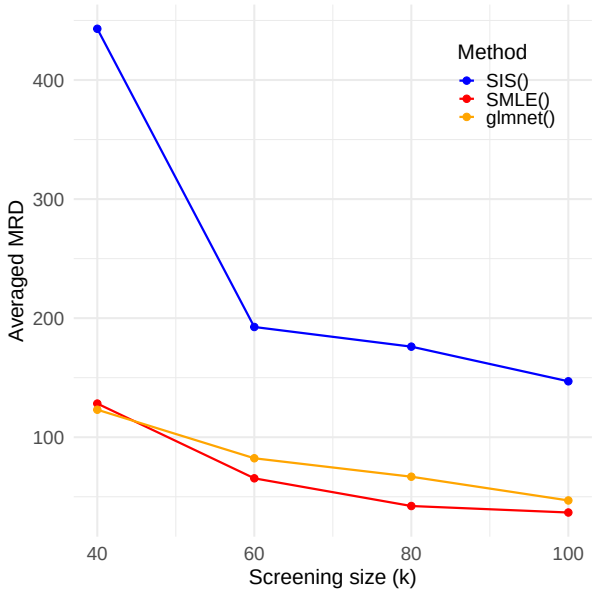


Figure 2.10: The averaged MRD on causal SNPs for `SIS()`, `glmnet()` and `SMLE()` with varying k.

2.5 Conclusion

In this chapter, we introduced **SMLE**, a user-friendly feature screening package designed to efficiently process ultrahigh-dimensional generalized linear models (GLMs). The package provides an implementation of the joint screening method SMLE, which yields more reliable screening results compared to commonly used marginal methods. Additionally, **SMLE** enables users to perform accurate post-screening selection through an accelerated Iterative Hard Thresholding (IHT) procedure, complemented by a preferred selection crite-

rion. Visualization tools are also included, allowing users to easily interpret screening and selection results for inference or prediction purposes. We demonstrated the usage of the main functions in **SMLE** with discussions and examples. The effectiveness and efficiency of the package are supported by extensive numerical studies. The idea of IHT has been demonstrated to be attractive in processing many types of ultrahigh-dimensional data. In the current version, the package focuses on feature screening for GLMs, which have been widely used in many scientific areas. It would be promising to extend the application scope of **SMLE** to other modeling scenarios such as Coxs proportional hazards models and multivariate response regression, where the IHT-based methodology has been discussed in the literature. For future work, it is also of great interest to embed **SMLE** in a distributed (federated) computing framework (Sun et al. [56]), so that it helps to conduct feature screening for big data that are stored in data segments. Depending on nature of a research problem, each data segment may measure the same set of features on different groups of objects, or measure different sets of features on the same group of objects. Due to legal, ethical or commercial restrictions, aggregating those segments is usually not allowed, and thus it is preferable to use **SMLE** in a distributed manner. Equipping **SMLE** with recent distributed estimation techniques (e.g. Jordan et al. [27]) provides a possible route to explore this emerging domain. In summary, **SMLE** stands out as a powerful tool for feature screening in ultrahigh-dimensional GLMs, offering reliable results and user-friendly functionalities. With future enhancements, including expanded modeling capabilities and distributed computing support, **SMLE** has the potential to become an even more versatile and indispensable resource for researchers and practitioners working with high-dimensional data.

Chapter 3

Stability-enhanced Feature Screening via Iterative Splicing for Sparse Estimation, with application to CLSA Metabolite Data

3.1 Introduction

In contemporary scientific research, ultrahigh-dimensional datasets containing an extensive number of features have become increasingly common. For instance, genetic researchers frequently examine hundreds of thousands to millions of genes to identify those associated with specific traits or diseases, while Internet firewalls process millions of keywords to evaluate security risks. Although these datasets offer substantial potential for uncovering

complex relationships, identifying predictive patterns, and advancing data-driven decision-making across various fields, they also pose significant challenges related to computational cost, statistical accuracy, and algorithmic stability when using traditional statistical methods [6, 9, 15, 16, 58, 89].

When the number of features is high, it is practical to assume that only a few are relevant. In regression analysis, this assumption translates to most predictors not affecting the response, a concept known as *sparsity*. A common strategy is to first employ *feature screening* to eliminate irrelevant features, which significantly simplifies the complexity of the data and reduces the dimensionality. This preprocessing step helps address computational challenges associated with high-dimensional datasets. The assumption that only a small portion of features are predictive has been widely adopted in high-dimensional models [13, 16, 58, 62, 67] and feature screening methods have been extensively studied in the literature [12, 30, 64, 88]. By focusing on identifying the most influential features, this approach enables the construction of interpretable models linking these features to the response variable.

While most screening methods focus on the marginal effects of features, they often neglect joint feature effects, as discussed earlier with approaches like Sure Independence Screening (SIS) [14], forward regression [61], and other model-based methods [16,88]. These techniques can miss important feature interactions when only weak marginal effects are present, leading to the exclusion of significant variables. To address this limitation, Fan and Lv [14] proposed the iterative application of Sure Independence Screening (ISIS), retaining fewer features in each round to capture joint effects. Similarly, Wang [61] recommended using classic forward regression for screening, which, although computationally

more intensive, better accounts for joint feature interactions. Xu and Chen [73] introduced an alternative joint screening approach using the Sparsity-restricted Maximum Likelihood Estimator (SMLE). Unlike traditional methods relying on an L_1 penalty, SMLE employs an L_0 penalty to limit the number of features. However, SMLE uses a local search strategy, and its performance is highly dependent on the initial values, often provided by a Lasso estimator. This reliance can limit SMLEs accuracy, as the quality of the initial estimates significantly impacts the methods effectiveness. Huang et al. [25] and Reese et al. [51] investigated feature screening methods for ultrahigh-dimensional categorical data. Barut et al. [3] proposed Conditional Sure Independence Screening (CSIS), a method designed to handle high-dimensional data by incorporating conditional relationships among variables. By conditioning on other selected variables, CSIS enhances its ability to select relevant predictors, making it more robust in capturing complex dependencies within the data. More recently, Zhu [87] introduced the Abess algorithm, an approach within the best-subset selection framework. Abess utilizes the gradient of the marginal likelihood to prioritize features based on their inner product with the residuals. However, this dynamic ranking can lead to overfitting when the number of retained features exceeds the number of causal features, as small residuals become difficult to capture. While Abess performs well with simple correlation structures, it faces challenges in more complex settings, as will be discussed in detail in Section 3.3.

We propose the Iterative Splicing for Sparse Estimation (ISSE), a method that offers a broader search range and remains stable regardless of initial values. ISSE addresses the challenge of high-dimensional feature screening, specifically the difficulty of identifying relevant features in the presence of a large number of variables and complex correlation

structures, by employing an iterative hard-thresholding algorithm on a restricted sparse model. In each iteration, we generate a candidate feature set by ranking the correlation between features with zero coefficients and the current residual. This candidate set is then merged with the active set, followed by re-estimating the restricted maximum likelihood. This process is repeated iteratively, with each step involving a neighborhood search strategy to approach the global optimum. The use of maximum likelihood estimation with an L_0 penalty ensures that both marginal and joint effects of features are thoroughly considered.

The remainder of this chapter is organized as follows: Section 3.2 introduces the ISSE methodology, providing an in-depth explanation of the algorithmic steps and the theoretical framework. Sections 3.3 and 3.4 present a comprehensive set of simulations for ISSE and adaptive ISSE, respectively. Section 3.5 discusses real-world experiments that highlight the advantages of ISSE over existing feature screening methods, especially when dealing with datasets with complex correlation structures. Finally, Section 3.6 summarizes the key findings and suggests potential directions for future research in feature screening technology.

3.2 Methodology

3.2.1 Sparse-Restricted Maximum Likelihood Estimator

Let $\boldsymbol{Y} = (y_1, \ldots, y_n)^\top$ denote the observed response variable for n samples, assumed to be constant over time. Consider a dataset with p features $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p\}$, where $\boldsymbol{x}_j = (x_{1j}, \ldots, x_{nj})^\top$ for $j = 1, \ldots, p$. Assume that some subset of these features causally

influences $\boldsymbol{Y}$. We follow the generalized linear model (GLM) framework as specified in Section 2.2.2:

$$f(\boldsymbol{Y};\theta) = \exp\{\theta\boldsymbol{Y} - b(\theta) + c(\boldsymbol{Y})\}, \tag{3.1}$$

where θ represents the natural parameter. The mean and variance are given by $E(\boldsymbol{Y}|\boldsymbol{X}) = b'(\theta)$ and $\text{Var}(\boldsymbol{Y}|\boldsymbol{X}) = b''(\theta)$, respectively, with θ linked to x through the relationship $g(\mu) = \boldsymbol{X}^\top\boldsymbol{\beta}$.

Recall the Sparse Maximum Likelihood Estimator (SMLE) defined in Section 2.2.2:

$$\widehat{\boldsymbol{\beta}}_k = \arg\max_{\boldsymbol{\beta}} l(\boldsymbol{\beta}) \quad \text{s.t.} \quad \|\boldsymbol{\beta}\|_0 \leq k, \tag{3.2}$$

where $\|\cdot\|_0$ denotes the L_0 norm, representing the number of non-zero elements. The SMLE aims to select k significant features while considering joint effects, similar to L_0-regularized approaches used in image processing.

Xu and Chen (2014) introduced an iterative hard-thresholding (IHT) algorithm to approximate the SMLE. The surrogate function

$$h(\boldsymbol{\beta},\boldsymbol{\gamma}) = l(\boldsymbol{\gamma}) + (\boldsymbol{\beta}-\boldsymbol{\gamma})^\top l'(\boldsymbol{\gamma}) - \frac{u}{2}\|\boldsymbol{\beta}-\boldsymbol{\gamma}\|_2^2 \tag{3.3}$$

is employed for its separability in $\boldsymbol{\beta}$. Starting with an initial value $\boldsymbol{\beta}^{(0)}$, a local solution is iteratively found through

$$\boldsymbol{\beta}^{(t+1)} = \arg\max_{\boldsymbol{\beta}} h(\boldsymbol{\beta},\boldsymbol{\beta}^{(t)}) \quad \text{s.t.} \quad \|\boldsymbol{\beta}\|_0 < k. \tag{3.4}$$

The unconstrained solution is given by

$$\tilde{\boldsymbol{\beta}}^{(t)} = \boldsymbol{\beta}^{(t)} + u^{-1}\boldsymbol{X}^{\top}[\boldsymbol{y} - b'(\boldsymbol{X}\boldsymbol{\beta}^{(t)})], \tag{3.5}$$

with $\boldsymbol{\beta}^{(t+1)} = H_k[\tilde{\boldsymbol{\beta}}^{(t)}]$, where H_k denotes the hard-thresholding operator that retains only the k largest components in absolute value.

3.2.2 Convergence Limitations in IHT and Feature Splicing for Adaptive Search

Given $\tilde{\boldsymbol{\beta}}$, the coefficient estimated during the IHT procedure, the estimated sparse pattern can be expressed by $\tilde{s} = \{j : \tilde{\beta}_j \neq 0, j = 1, .., p\}$. When the algorithm has converged, the nonzero elements in $\tilde{\boldsymbol{\beta}}$ are unchanged; therefore, from Xu and Chen [73], we can deduce that $\partial h_n(\boldsymbol{\gamma}, \tilde{\boldsymbol{\beta}})/\partial\gamma_j|_{\boldsymbol{\gamma}=\tilde{\boldsymbol{\beta}}} = u^{-1}\boldsymbol{x}_j^{\top}(\boldsymbol{y} - b'(\boldsymbol{X}\tilde{\boldsymbol{\beta}})) = 0$ for $j \in \tilde{s}$, where $\boldsymbol{x}_j$ is the jth column of $\boldsymbol{X}$. This further implies that $[\partial l_n(\boldsymbol{\beta})/\partial\boldsymbol{\beta}]_j = 0$ at $\boldsymbol{\beta} = \tilde{\boldsymbol{\beta}}$ for $j \in \tilde{s}$. Hence, in the space of $\tilde{\boldsymbol{\beta}}$ of this sparsity pattern, $\tilde{\boldsymbol{\beta}}$ is the unique maximum because $l_n(\cdot)$ is strictly convex.

It is important to note that $\tilde{\boldsymbol{\beta}}$ represents a local maximum of $l_n(\boldsymbol{\beta})$. However, in highly correlated scenarios, it can be difficult to ensure that $s^* = \{j : \beta_j^* \neq 0, j = 1, \ldots, p\} \subseteq \tilde{s}$. The Sure Screening Property may not hold in all scenarios, and SMLE can fail due to the complexity of the data structure or poor initialization.

To enhance the screening accuracy of the Sparse Maximum Likelihood Estimation (SMLE), we conduct an in-depth analysis for cases where $s^* \not\subseteq \tilde{s}$. Remarkably, we observe that not only are the nonzero values in $\tilde{\boldsymbol{\beta}}$ stable, but the subset $\tilde{s}$ itself also demonstrates

stability. This stability is evident from the following inequality:

$$\max\left\{\left|u^{-1}\boldsymbol{x}_j^\top(\boldsymbol{y}-b'(\boldsymbol{X}\tilde{\boldsymbol{\beta}}))\right| : j \notin \tilde{s}\right\} < \min\left\{\left|\tilde{\beta}_j\right| : j \in \tilde{s}\right\}, \tag{3.6}$$

which holds when the IHT algorithm has converged to a solution. Specifically, if there exists any $j \notin \tilde{s}$ such that

$$\left|u^{-1}\mathbf{x}_j^\top(\mathbf{y}-b'(\mathbf{X}\tilde{\boldsymbol{\beta}}))\right| > \min\left\{\left|\tilde{\beta}_i\right| : i \in \tilde{s}\right\},$$

then during the hard-thresholding step, the feature corresponding to index j will be added to the set $\tilde{s}$, while the feature associated with the smallest coefficient in $\tilde{s}$ will be removed. Consequently, the IHT algorithm will continue to iterate and will not reach convergence at this step. So this inequality holds at equilibrium.

From inequality (3.6), we can deduce two reasons why $s^* \not\subseteq \tilde{s}$:

1. The step size u^{-1} is too small as $\boldsymbol{\beta}^{(t)}$ converges towards $\tilde{\boldsymbol{\beta}}$, potentially affecting the dynamic range of updates and the convergence behavior.

2. For indices $j \in s^* \setminus \tilde{s}$, the magnitude of $\tilde{\rho}_j := \boldsymbol{x}_j^\top(\boldsymbol{y}-b'(\boldsymbol{X}\tilde{\boldsymbol{\beta}}))$ is insufficient to meet the threshold required for inclusion in $\tilde{s}$.

The magnitude $|\tilde{\rho}_j|$ serves as a metric to gauge the correlation between the regression residuals and the features not selected in the model $\tilde{s}$. For instance, in a linear model, $\tilde{\rho}_j = \boldsymbol{x}_j^\top(\boldsymbol{y}-\boldsymbol{X}\tilde{\boldsymbol{\beta}})$ quantifies this correlation. Considering scenarios where all relevant features cannot be adequately represented by a few irrelevant predictors, the exclusion

of crucial predictors leads to a significant correlation of the remaining relevant predictors with the residual. This concept is supported by methodologies such as ISIS by Fan [15] and Abess by Zhu [87] in linear models.

To avoid missing relevant features due to this phenomenon, we can implement an adaptive step size in the IHT algorithm. Specifically, we should choose a larger step size to escape local search regions and ensure an increase in the likelihood. From inequality (3.6), either a larger step size or an increase in $\tilde{\rho}_j$ can facilitate this update. We propose two potential factors contributing to the small $\tilde{\rho}_j$ for $j \in s^* \setminus \tilde{s}$. The first factor is the strong correlation between $\boldsymbol{X}_j$ for $j \in s^* \setminus \tilde{s}$ and $\boldsymbol{X}_j$ for $j \in \tilde{s}$, and the second is the presence of many redundant irrelevant features retained in $\tilde{s}$. Additionally, a very small value of β_j^* for $j \in s^* \setminus \tilde{s}$ may be a contributing factor, though this is not considered for now.

To facilitate the inclusion of relevant but previously excluded features, we employ the metric $|\tilde{\rho}_j|$, which quantifies the residual impact of these features. For simplicity, denote the complement of the feature set s in the model as $\tilde{s}^c = \{1, \ldots, p\} \setminus s$. Within this subset, a specific submodel defined by s is evaluated using the Maximum Likelihood Estimator (MLE), denoted by $\widehat{\boldsymbol{\beta}}$ or $\widehat{\boldsymbol{\beta}}(s)$. This estimator is foundational for establishing the selection criteria in our feature inclusion strategy.

The key subsets for consideration are defined as follows:

$$
\begin{aligned}
\mathcal{A}_h(s) &= \left\{ j : |\widehat{\beta}_j| \text{ is among the top } h \text{ largest values}, j \in s \right\}, \\
\mathcal{B}_h(s) &= \left\{ j : |\boldsymbol{x}_j^\top (\boldsymbol{y} - b'(\boldsymbol{X}\widehat{\boldsymbol{\beta}}))| \text{ is among the top } h \text{ largest values}, j \in s^c \right\}, \\
\mathcal{C}_h(s) &= \left\{ j : |\boldsymbol{x}_j^\top (\boldsymbol{I}_n - \boldsymbol{\Pi}_s)(\boldsymbol{y} - b'(\boldsymbol{X}\widehat{\boldsymbol{\beta}}))| \text{ is among the top } h \text{ largest values}, j \in s^c \right\},
\end{aligned}
$$

where $\boldsymbol{I}_n$ is the identity matrix of size n, and $\boldsymbol{\Pi}_s = \boldsymbol{X}_s(\boldsymbol{X}_s^\top \boldsymbol{X}_s)^{-1}\boldsymbol{X}_s^\top$ is the projection matrix onto the column space of $\boldsymbol{X}_s$.

$\mathcal{A}_h(s)$ describes the potentially relevant features based on their marginal coefficients. This approach is straightforward, but its drawback is that the selected features may be correlated with those already in the active set, making them less valuable for improving the model. On the other hand, $\mathcal{B}_h(s)$ selects the top h features most correlated with the residual. Adding these features can reduce prediction error and improve the model. However, ρ_j (the correlation between the feature and the residual) may exhibit collinearity with the column space of $\boldsymbol{X}_s$. Inspired by the High-dimensional Ordinary Least-Squares Projection (HOLP) [66], we rank features by their independent correlation with the residual to mitigate this issue, and thus construct $\mathcal{C}_h(s)$.

When the IHT procedure terminates, the set $\tilde{s}$ may not fully contain s^*. To address this, we introduce a pre-fixed parameter h to select up to h relevant features from $\tilde{s}^c$ using $\mathcal{A}_h(s)$, $\mathcal{B}_h(s)$ or $\mathcal{C}_h(s)$. The updated active set is then obtained by adding elements from either $\mathcal{A}_h(s)$, $\mathcal{B}_h(s)$, or $\mathcal{C}_h(s)$ to $\tilde{s}$, such that:

$$\tilde{s}^{\text{new}} = \tilde{s} \cup \mathcal{X}_h(s),$$

where $\mathcal{X}_h(s)$ is one of $\mathcal{A}_h(s)$, $\mathcal{B}_h(s)$, or $\mathcal{C}_h(s)$, depending on the chosen scenario. Once $\tilde{s}^{\text{new}}$ is constructed, we reapply IHT with a restricted screening problem using the non-zero elements in $\tilde{s}^{\text{new}}$. This can be treated as an improvement over a random initial value or a boundary-searching strategy. We call this strategy Iterative Splicing, and the algorithm is described in Algorithm 3.1.

Algorithm 3.1 Simplified ISSE Algorithm for Feature Screening

Input: Response vector Y, Predictor matrix X, Sparsity level k, Initial coefficients $\boldsymbol{\beta}_0$
Parameters: Family of the GLM, Splicing size h
Initialize: Initial coefficients $\boldsymbol{\beta}_0$, feature set s_0
Perform initial Iterative Hard Thresholding (IHT): $\tilde{\boldsymbol{\beta}}_t = \text{IHT}(\boldsymbol{\beta}_t)$
Repeat until likelihood does not increase:

- Determine screening set $\mathcal{X}_h(s)$
- Update the active set: $\tilde{s}^{\text{new}} = \tilde{s} \cup \mathcal{X}_h(s)$
- Update coefficients using GLM based on $\tilde{s}^{\text{new}}$: $\widehat{\boldsymbol{\beta}}^{\text{new}}$
- Apply IHT on $\widehat{\boldsymbol{\beta}}^{\text{new}}$ to get $\tilde{\boldsymbol{\beta}}_{t+1} = \text{IHT}(\widehat{\boldsymbol{\beta}}^{\text{new}})$
- Compute likelihood based on $\tilde{\boldsymbol{\beta}}_{t+1}$
- Identify s_t based on $\tilde{\boldsymbol{\beta}}_{t+1}$

Output: Final optimized coefficients $\widehat{\boldsymbol{\beta}}_{\text{end}}$ and selected feature set $\widehat{s}_{\text{end}}$

3.2.3 Adaptive Iterative Splicing Screening

In practice, the support size s is typically unknown. To estimate s, we employ a data-driven approach. Information criteria, such as the high-dimensional Bayesian Information Criterion (HBIC) [63] and the Extended Bayesian Information Criterion (EBIC) [7], are frequently utilized for this purpose. While HBIC [63] is primarily used to select tuning parameters in penalized likelihood estimation, the focus of this work is on recovering the support size s for screening using EBIC.

For a given active set A, EBIC is defined as:

$$\text{EBIC}(A) = -2\log L_A + |A|\log(n) + 2\gamma|A|\log(p),$$

where L_A is the likelihood for the active set A, n is the sample size, p is the number of predictors, and $\gamma \in [0, 1]$ is a parameter controlling the penalty for model complexity. This criterion provides a balance between model fit and complexity, with the additional term $2\gamma|A|\log(p)$ accounting for high-dimensional settings.

Unlike the standard ISSE algorithm, the adaptive ISSE (aISSE) algorithm estimates the current model in each iteration. Once a local maximum is identified, the best model is selected based on the lowest Extended Bayesian Information Criterion (EBIC). The estimated model is then denoted as $\widehat{s}$, and the key sets are constructed as $\mathcal{X}_h(\widehat{s})$. The corresponding updated splicing set is defined as:

$$\tilde{s}^{\text{new}} = \tilde{s} \cup \mathcal{X}_h(\widehat{s}),$$

after which the IHT algorithm is applied with the new sparsity level $\widehat{s}$. This procedure is summarized in Algorithm 3.2.

Algorithm 3.2 Adaptive ISSE Algorithm for Feature Screening

Input: Response vector Y, predictor matrix X, initial coefficients $\boldsymbol{\beta}_0$
Parameters: Generalized Linear Model (GLM) family, splicing size h, penalty parameter γ for EBIC
Initialize: Initial coefficients $\boldsymbol{\beta}_0$, initial feature set s_0
Perform initial Iterative Hard Thresholding (IHT): $\tilde{\boldsymbol{\beta}}_0 = \text{IHT}(\boldsymbol{\beta}_0)$
Repeat until the likelihood converges:

- Compute the model with the lowest EBIC:

$$\widehat{s} = \arg\min \text{EBIC}(s) = -2 \log L_s + |s| \log(n) + 2\gamma |s| \log(p),$$

  where L_s is the likelihood for the model with active set s.

- Identify the screening set: $\mathcal{X}_h(\widehat{s})$
- Update the active set: $\tilde{s}^{\text{new}} = \tilde{s} \cup \mathcal{X}_h(\widehat{s})$
- Update the coefficients based on the new active set using the GLM: $\widehat{\boldsymbol{\beta}}^{\text{new}} = \text{GLM}(\tilde{s}^{\text{new}})$
- Apply IHT to $\widehat{\boldsymbol{\beta}}^{\text{new}}$ to refine the coefficients: $\tilde{\boldsymbol{\beta}}_{t+1} = \text{IHT}(\widehat{\boldsymbol{\beta}}^{\text{new}} | \widehat{s})$
- Recompute the likelihood based on the updated coefficients $\tilde{\boldsymbol{\beta}}_{t+1}$

Output: Final optimized coefficients $\widehat{\boldsymbol{\beta}}_{\text{end}}$ and the selected feature set $\widehat{s}_{\text{end}}$

3.3 Simulation Studies

In our simulation studies, we evaluated the performance of the ISSE method and compared it with other feature screening techniques across a range of scenarios. Our primary focus

was on each methods' ability to identify relevant features, its predictive accuracy, and computational efficiency as measured by computational time. The effectiveness of each approach was evaluated using several metrics.

We assess each methods ability to identify relevant features using the positive screening rate and the sure screening property. The positive screening rate (PSR) is

$$\text{PSR} = \frac{|S \cap \widehat{S}|}{|S|}$$

where S is the set of true relevant features, and $\widehat{S}$ is the set of features identified by the screening method. The sure screening rate is (SSR),

$$\text{SSR} = \begin{cases} 1, & \text{if } S \subseteq \widehat{S}, \\ 0, & \text{if } S \not\subseteq \widehat{S}. \end{cases}$$

We assess predictive accuracy using test loss functions specific to each model type. For the Gaussian model, test loss was calculated as the sum of the absolute differences between predicted values and actual test labels.

$$\text{Test Loss}_{\text{Gaussian}} = \sum_{i=1}^{n} (y_i - \widehat{y}_i)^2$$

In the case of the Poisson model, we employed log loss to quantify test loss:

$$\text{Test Loss}_{\text{Poisson}} = -\sum_{i=1}^{n} (y_i \log(\widehat{y}_i) - \widehat{y}_i)$$

For the Binomial model, test loss was measured using the logit loss:

$$\text{Test Loss}_{\text{Binomial}} = -\sum_{i=1}^{n} \left(y_i \log(\widehat{y}_i) + (1 - y_i) \log(1 - \widehat{y}_i)\right)$$

3.3.1 Evaluating ISSE Performance Across Diverse Models and Correlation Structures

In our initial simulation study, we aim to assess the performance of the ISSE method across various model types and correlation structures. To this end, we designed three distinct scenarios: an autoregressive correlation pattern, a block-wise correlation structure, and a cyclical correlation structure. For each data structure, we considered three models and two levels of noise.

Autoregressive Correlation Structure

The autoregressive (AR) correlation structure is characterized by a parameter ρ. This setting indicates that the value of a given feature is influenced by its preceding feature's value, establishing a correlation pattern wherein features in closer proximity exhibit a stronger correlation compared to those farther apart. The covariance between any two features i and j, denoted by Σ_{ij}, is defined as $\rho^{|i-j|}$, with $\rho = 0.8$. This formulation embodies the AR(1) process's principle, indicating that the correlation weakens as the distance between features increases.

In our simulation, we consider three distinct settings. In Setting 1, the sample size N is 200, the number of features P is 4000, and the number of causal features k is 20, with the distribution family set to Gaussian. The causal features are positioned at

$\{101, 103, 105, 107, 109, 111\}$, and their coefficients alternate in sign, scaled by a factor of 2: $2 \times \{1, -1, 1, -1, 1, -1\}$. In Setting 2, the sample size increases to $N = 350$, while the number of features P, the number of causal features k, and the positions of the causal features remain unchanged. However, the distribution family is Binomial, with the same alternating coefficients as in Setting 1. Finally, in Setting 3, the sample size is $N = 400$, with $P = 4000$ and $k = 20$ as in the previous settings. The distribution family is Poisson, and while the positions of the causal features remain the same, the coefficients are scaled by 0.8: $0.8 \times \{1, -1, 1, -1, 1, -1\}$.

Table 3.1: Simulation Results of Gaussian, Logistic and Poisson Models with Autoregressive Correlation Structure.

Model Type	Method	Low Noise Level				Moderate Noise Level			
		SSR	PSR	Test Loss	Time	SSR	PSR	Test Loss	Time
G	SIS	0.00	0.48	4.23	0.06	0.00	0.47	5.72	0.07
	(I)SIS	0.32	0.65	3.84	5.23	0.00	0.24	5.86	7.22
	Lasso	0.50	0.85	1.55	0.02	0.00	0.60	5.05	0.02
	SMLE	1.00	1.00	0.02	0.07	1.00	1.00	1.46	0.08
	Abess	0.90	0.95	0.59	0.02	0.70	0.85	3.23	0.02
	ISSE	1.00	1.00	0.01	0.11	1.00	1.00	1.53	0.11
L	SIS	0.00	0.45	0.71	0.16	0.00	0.44	0.73	0.17
	(I)SIS	0.00	0.52	0.76	18.77	0.00	0.46	0.86	18.40
	Lasso	0.00	0.46	0.79	0.03	0.00	0.45	0.81	0.03
	SMLE	0.56	0.88	0.71	0.16	0.43	0.83	0.77	0.17
	Abess	0.12	0.62	1.21	0.08	0.16	0.63	1.25	0.09
	ISSE	0.92	0.97	0.73	0.14	0.90	0.97	0.77	0.15
P	SIS	0.00	0.53	2.05	0.20	0.00	0.43	2.36	0.21
	(I)SIS	0.00	0.53	2.13	28.94	0.00	0.43	2.86	26.21
	Lasso	0.00	0.53	2.16	0.03	0.00	0.43	2.36	0.03
	SMLE	0.90	0.98	1.54	0.24	0.73	0.94	1.87	0.28
	Abess	0.93	0.98	1.66	0.14	0.57	0.84	2.15	0.11
	ISSE	1.00	1.00	1.61	0.27	0.90	0.97	1.92	0.29

To assess the performance of feature screening methods, we ran 100 repeated simulations across different autoregressive (AR) scenarios. This repetition ensures the results are reliable and robust, offering a thorough comparison of methods. The simulation results in Table 3.1 demonstrate that the ISSE method consistently outperforms other feature screening techniques, especially in low and moderate noise conditions for the Logistic model. ISSE achieves the highest positive screening rate (PSR) and sure screening rate (SSR) while maintaining competitive test loss and computational time. In the Gaussian model, ISSE delivers perfect SSR and PSR values of 1.00, correctly identifying all relevant features without errors, and achieves the lowest test loss (0.01 and 1.53 for low and moderate noise, respectively), surpassing SMLE. For the Poisson model, ISSE maintains a strong SSR (1.00 and 0.90) and PSR (1.00 and 0.97), with test losses comparable to SMLE but superior feature identification. Although ISSE's computational time is slightly higher, its accuracy and feature screening reliability make this a reasonable trade-off. Other methods, like SIS and Lasso, often struggle with consistency under moderate noise, while SMLE and Abess offer competitive performance. Overall, ISSE is the most reliable and effective method, excelling in feature screening accuracy, predictive performance, and robustness to noise, justifying its slightly higher computational cost.

Block Wise Correlation Structure

In genetic research, genomes are partitioned into segments or regions delineated by linkage disequilibrium (LD), which quantifies the non-random association of alleles across different loci. Each LD block is characterized by alleles that are co-inherited more frequently than would be expected by chance, signifying a strong interconnection between genetic variants within the same block. However, the correlation between sites across

distinct blocks is typically low. This structure justifies the assumption of independence between certain regions in analytical models.

Our simulation efforts aim to mimic datasets with block-like correlation structures. To achieve blockwise correlation structure, we organize the data into blocks using a specified correlation matrix, $\Sigma = \text{diag}\{\Sigma_1, \Sigma_2\}$, where each Σ_m represents submatrices defined by $\rho_1 = 0.7$ and $\rho_2 = 0.8$. These blocks reflect high correlation within the same block and low correlation between distinct blocks. We then transform the aggregated continuous responses into probabilities or lambda values, which are used to generate binary outcomes through a Bernoulli distribution or integer outcomes using a Poisson distribution.

We establish three unique simulation settings to assess performance across diverse scenarios. In the first setting, a Gaussian distribution is used with sample size $N = 650$, total number of features $P = 8000$, and pre-fixed model size $k = 40$. The 12 causal feature positions are $\{101, \ldots, 113, 2101, \ldots, 2113\}$ with alternating sign coefficients scaled by 2: $2 \times \{1, -1, \ldots, -1\}$. In the second setting, we utilize a Binomial distribution family with $N = 900$, $P = 8000$, and $k = 20$. The locations and coefficients of the causal features mirror those in Setting 1. In the third setting, under a Poisson distribution family, we use $N = 650$, $P = 8000$, and $k = 40$. The positions of the causal features remain consistent with the earlier settings, but the coefficients are adjusted to a scale of 1. The changes in N and k are made because all methods in the table require different levels of information to achieve optimal performance under different models.

Table 3.2: Simulation Results of Gaussian, Logistic and Poisson Models with Block wise Correlation Structure.

Model Type	Method	Low Noise Level				Moderate Noise Level			
		SSR	PSR	Test Loss	Time	SSR	PSR	Test Loss	Time
Gaussian	SIS	0.00	0.36	4.42	0.05	0.00	0.48	5.97	0.05
	(I)SIS	0.62	0.85	1.52	5.42	0.34	0.69	5.27	7.24
	Lasso	0.98	0.65	1.33	0.02	0.65	0.90	1.59	0.02
	SMLE	1.00	1.00	0.01	0.06	0.98	0.99	1.60	0.07
	Abess	0.99	0.98	0.11	0.02	0.68	0.84	3.23	0.02
	ISSE	1.00	1.00	0.01	0.06	1.00	1.00	1.53	0.08
Logistic	SIS	0.00	0.36	0.62	0.17	0.00	0.36	0.63	0.17
	(I)SIS	0.02	0.52	0.60	5.77	0.00	0.46	2.86	5.40
	Lasso	0.59	0.92	0.60	0.05	0.50	0.90	0.61	0.05
	SMLE	0.39	0.88	0.69	0.31	0.36	0.87	0.71	0.30
	Abess	0.65	0.93	0.73	0.16	0.49	0.89	0.73	0.16
	ISSE	0.66	0.92	0.59	0.18	0.58	0.91	0.60	0.16
Poisson	SIS	0.00	0.39	4.22	0.49	0.00	0.38	4.79	0.49
	(I)SIS	0.03	0.52	2.23	15.28	0.01	0.42	4.26	15.29
	Lasso	1.00	1.00	1.40	0.19	0.98	0.99	1.99	0.17
	SMLE	1.00	1.00	1.45	0.37	1.00	1.00	2.13	0.45
	Abess	0.97	0.99	1.55	0.34	0.94	0.98	2.15	0.35
	ISSE	1.00	1.00	1.40	0.51	1.00	1.00	1.98	0.62

The simulation results in Table 3.2 show that the ISSE method consistently outperforms other feature screening techniques across Gaussian, Logistic, and Poisson models with block-wise correlation structures. ISSE achieves the highest sure screening rate (SSR) and positive screening rate (PSR) in most cases, particularly under low noise, with perfect SSR and PSR values of 1.00 for both Gaussian and Poisson models. Despite slightly higher computational times compared to Lasso and Abess, ISSE maintains competitive test losses and provides superior feature identification, especially under moderate noise. Its robust

performance in predictive accuracy and feature selection justifies the marginally increased computational cost, making ISSE a highly reliable method in high-dimensional feature screening tasks.

Cyclical Period Correlation

The cyclical period correlation mimics real-world phenomena where observable patterns are not static but fluctuate in a predictable manner over time. A cosine function, modulated by an amplitude (a) and a cyclical period (T), it captures the essence of these fluctuations within the correlation matrix Σ, where each element, Σ_{ij}, is defined as:

$$\Sigma_{ij} = \begin{cases} 1 & \text{if } i = j, \\ a \cdot \cos\left(\frac{2\pi \cdot |i-j|}{T}\right) & \text{otherwise,} \end{cases}$$

We use a cyclical period of $T = 2000$ and an amplitude $a = 0.9$. In Setting 1, we set sample size $N = 250$, total number of features $P = 1000$, and prefix model size $k = 15$, with a Gaussian distribution. The causal features are strategically positioned at intervals reflecting key positions: $\{1, 500, 1000, 1500, 2000\}$, each with coefficients that reflect a patterned influence: $\{2, -2, 2, -2, 1\}$. Setting 2 uses $N = 300$, $P = 1000$, and $k = 15$, under a Binomial distribution. This setup mirrors the causal feature positions of Setting 1 but adopts a uniform coefficient value of 2, to examine the consistent influence across different distributions. In Setting 3, with $N = 350$, $P = 1000$, and $k = 15$, following a Poisson distribution, the positions of causal features remain consistent with prior configurations, while the coefficients are adjusted to $\{1.5, -1.5, 1.5, -1.5, 1.5\}$; this choice allows for an evaluation of the effect of coefficient magnitude under a cyclical correlation structure.

Table 3.3: Comparative Performance Metrics for Poisson, Gaussian, and Logistic Models under Cyclical Period Correlation.

Model Type	Method	Low Noise Level				Moderate Noise Level			
		SSR	PSR	Test Loss	Time	SSR	PSR	Test Loss	Time
Gaussian	SIS	0.00	0.36	1.71	0.04	0.00	0.33	2.54	0.05
	(I)SIS	0.66	0.85	0.84	5.77	0.27	0.69	2.86	6.40
	Lasso	0.98	1.00	0.36	0.02	0.52	0.89	1.59	0.02
	SMLE	0.98	1.00	0.45	0.06	0.51	0.90	1.92	0.07
	Abess	0.89	0.98	0.41	0.02	0.23	0.78	2.29	0.02
	ISSE	1.00	1.00	0.36	0.08	0.68	0.92	1.53	0.08
Logistic	SIS	0.00	0.38	0.62	0.24	0.00	0.37	0.63	0.22
	(I)SIS	0.02	0.52	0.60	5.77	0.00	0.46	2.86	5.40
	Lasso	0.74	0.95	0.58	0.11	0.74	0.95	0.59	0.11
	SMLE	0.57	0.91	0.66	0.51	0.51	0.90	0.66	0.52
	Abess	0.78	0.96	0.67	0.20	0.75	0.95	0.68	0.20
	ISSE	0.79	0.96	0.57	0.30	0.71	0.94	0.59	0.29
Poisson	SIS	0.00	0.39	4.04	0.38	0.00	0.37	4.68	0.40
	(I)SIS	0.72	0.89	3.13	28.94	0.23	0.66	2.86	20.69
	Lasso	1.00	1.00	1.40	0.08	0.95	0.99	1.97	0.07
	SMLE	1.00	1.00	1.45	0.19	0.99	1.00	2.10	0.22
	Abess	0.98	1.00	1.47	0.26	0.92	0.97	2.19	0.25
	ISSE	1.00	1.00	1.41	0.31	0.97	0.99	1.98	0.37

The simulation results in Table 3.3 highlight the performance of the ISSE method compared to other feature screening techniques under a cyclical period correlation structure across Poisson, Gaussian, and Logistic models. ISSE consistently achieves a high sure screening rate (SSR) and positive screening rate (PSR), particularly under low noise levels, where it attains perfect SSR and PSR (1.00) for both Gaussian and Poisson models. Even under moderate noise, ISSE maintains strong performance, with an SSR of 0.68 for the Gaussian model and 0.97 for the Poisson model, outperforming methods like SIS and (I)SIS, which struggle with feature identification under moderate noise. While the computational time for ISSE is slightly higher than methods like Lasso and Abess, its improved accuracy in both test loss and feature screening justifies the trade-off. Overall, ISSE proves to be a robust and reliable method across all models and noise levels.

3.3.2 Simulations with Varying Sample Sizes, Model Complexities, and Feature Dimensions in High-Dimensional Binary Data

In our second simulation study, we aimed to examine the characteristics of various screening methods by generating datasets with progressively increasing sample sizes, model complexities, and feature dimensions. These datasets were created with causal features located at positions 101, 103, 105, 107, 109, and 111, and with each assigned predetermined coefficients. We assumed a binomial distribution for the outcome variable and an autoregressive (AR) pattern to simulate correlations among features, giving a challenging setup for the evaluation of screening methods. The simulation was structured to increase the sample

size incrementally, starting from 150 and ending at 800, conducting 100 simulations at each level. Similarly, the pre-fixed model size was increased incrementally, starting from 5 and ending at 28, conducting 100 simulations at each level. Additionally, the data dimension was varied incrementally, starting from 1500 and increasing to 6000. This step was intended to evaluate how each screening method scales with higher feature dimensions, in order to examine the computational challenges and prediction accuracy in the presence of a larger number of features.

Figure 3.1 compares five feature screening methods – ISSE, SMLE, Abess, SIS, and Lasso – across three experimental conditions: increasing the number of features, increasing the sample size, and increasing model sparsity. The results are measured using three metrics: Sure Screening Rate (SSR), Positive Screening Rate (PSR), and Test Error. In the top row, as the number of features increases from 1500 to 6000, SMLE and ISSE demonstrate superior performance in retaining relevant features (higher SSR and PSR) while maintaining low test error rates compared to SIS and Lasso. Notably, the performance of other methods declines significantly with increasing dimensionality, whereas ISSE maintains consistently high performance. In the middle row, as the sample size increases from 150 to 800, ISSE and SMLE consistently outperform other methods in feature screening accuracy. However, ISSE shows a particularly strong SSR when the sample size is low. The bottom row, which examines the effect of model sparsity, reveals that ISSE and SMLE remain highly effective at identifying relevant features across varying sparsity levels, while Lasso and SIS exhibit relatively poor performance under all conditions. Furthermore, Abess's performance declines significantly with increasing model complexity, highlighting its limitations in feature screening. This comparison highlights the robustness of ISSE across

various feature dimensions, sample sizes, and sparsity levels, positioning it as the most reliable method for feature screening in this simulation study.

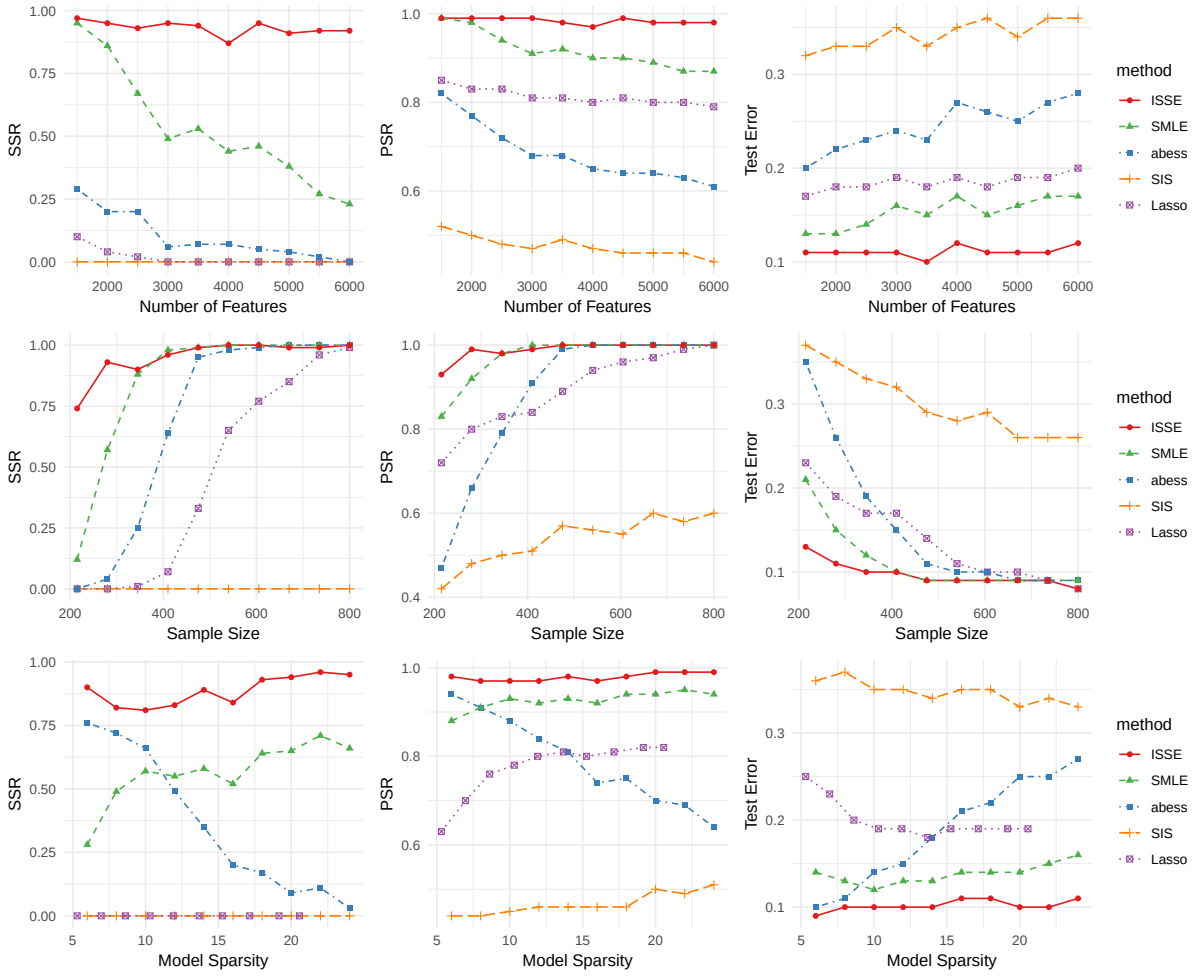


Figure 3.1: Comparison of five feature screening methods (ISSE, SMLE, Abess, SIS, Lasso) under varying sample sizes, model complexities, and feature dimensions. All datasets were simulated under a binary model.

3.3.3 Comparison of ISSE, SMLE and Abess in Terms of Likelihood

In this simulation study, we aimed to evaluate the performance of the Abess, SMLE and ISSE methods for sparse model selection in high-dimensional settings to demonstrate that ISSE has a higher likelihood of retaining a better model. We generated synthetic datasets with $N = 200$ samples and $P = 2000$ features. The features were drawn from a multivariate normal distribution with an AR(1) correlation structure, defined by the parameter $\rho = 0.8$. A sparse true model was specified with only 6 causal features located at indices $101, 103, 105, 107, 109, 111$, with alternating coefficients of 2 and -2. We simulated 100 datasets and for each dataset. ISSE, SMLE and Abess methods were applied to identify the subset of features that best explained the response variable. The support size for both methods was set to $k = 15$. We returned the log-likelihood of the models fitted by Abess, SMLE and ISSE.

Log-likelihood values for a single simulation are shown in Figure 3.2. The model coefficients were initialized to 0, resulting in a low initial log-likelihood. The figure illustrates six steps of ISSE splicing, with each splicing iteration increasing the log-likelihood of the model. After the first splicing, the ISSE model surpasses the log-likelihood values achieved by the Abess and SMLE methods, demonstrating the efficiency of splicing for this problem.

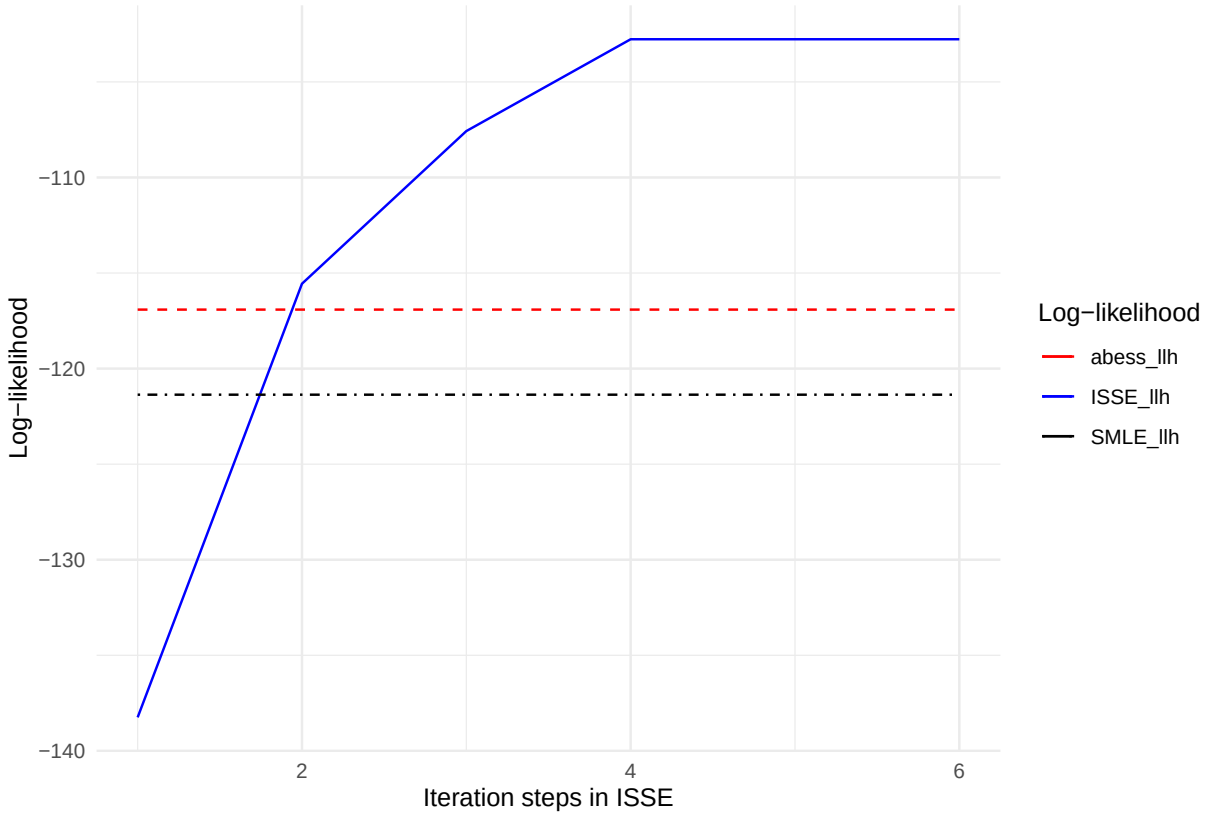


Figure 3.2: ISSE log-likelihood curve over iterations (blue line). For comparison, the maximum log-likelihood values for the Abess and SMLE models are depicted in red and black, respectively.

Figure 3.3 summarizes the log-likelihood values across the 100 simulated datasets. The figure illustrates the distribution of log-likelihood values for the models fitted using the `Abess`, `SMLE`, and `ISSE` methods. The boxplots show that the `ISSE` method generally achieves higher log-likelihood values compared to the `Abess` and `SMLE` method, indicating better model fit.

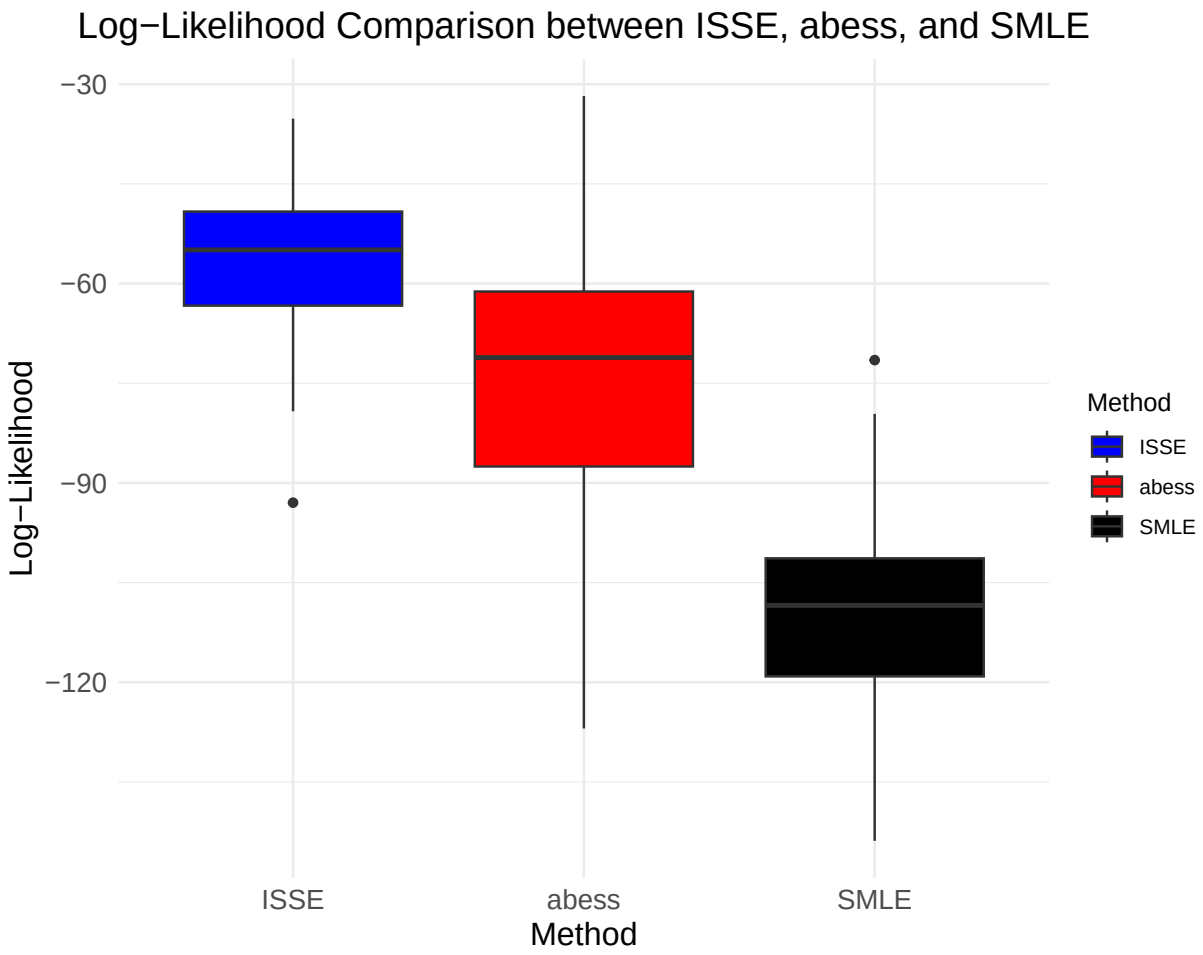


Figure 3.3: Box plot comparing the log-likelihood values of the models fitted by `ISSE`, `Abess` and `SMLE` methods.

3.4 Simulation Study for Adaptive Iterative Splicing Sparse Estimation

One drawback of ISSE is the requirement for prior knowledge about the dataset, including the target model size k (Figure 3.1). Although the performance of ISSE is not highly sensitive to the choice of k, we introduce the Extended Bayesian Information Criterion (EBIC) to help control the model size and adaptively select the appropriate model.

To evaluate the adaptive choice of k (aISSE), we simulated 100 datasets using an autoregressive (AR) correlation structure, following the specifications of Scenario 2, with a sample size of $N = 500$ and $P = 7000$ features. The distribution family was set to binomial, and six causal features were positioned at indices $\{101, 103, 105, 107, 109, 111\}$. For a more fair comparison, we compared the performance of aISSE to the analogous adaptive versions of Abess (Abess Select), SMLE (SMLE-EBIC), SIS (SIS-S) and Lasso (Lasso-CV). The SIS method was implemented using the R package [53], selecting indices with the default SCAD regularization step. Lasso was performed with the R package [58], with the regularization parameter λ chosen through cross-validation. SMLE-EBIC utilized the R package, with `selection = TRUE` and the default criterion set to `EBIC`. The Abess method was applied using the default settings from the **Abess** package [86]. All methods were implemented using publicly available R packages.

The results, presented in Table 3.4, compare the performance of five feature selection methods: SIS-S, Lasso-CV, SMLE-EBIC, Abess Select, and aISSE. Performance is evaluated based on the average True Positive Rate (TPR), False Discovery Rate (FDR), Test Error, Sparsity Level Estimate (SLE), and Model Size across 100 simulated datasets. The

Sparsity Level Estimate (SLE) is defined as SLE $= \big||S| - |\widehat{S}|\big|$, where S represents the set of true relevant features, $\widehat{S}$ denotes the set of features identified by the screening method, and $|S|$ denotes the cardinality of the set.

Table 3.4: Average performance comparison of feature selection methods

Method	TPR	FDR	Model Size	SLE	Test Error
aISSE	**0.98**	0.15	7.02	1.12	**0.51**
Abess Select	0.94	0.17	7.08	1.52	0.53
SMLE-EBIC	0.89	**0.05**	5.70	**0.86**	0.52
SIS-S	0.40	0.48	7.59	4.83	0.67
Lasso-CV	0.44	0.49	**12.52**	9.78	1.07

The results highlight the superiority of aISSE, which achieves the highest TPR (0.98), demonstrating its superior ability to recover true causal features. This makes it the most accurate method for feature selection, outperforming Abess Select (TPR = 0.94) and SMLE-EBIC (TPR = 0.89). Furthermore, aISSE also achieves the lowest Test Error (0.51) while maintaining a relatively small model size (7.02), striking a balance between feature selection accuracy and model complexity.

Although SMLE-EBIC records the lowest FDR (0.05) and smallest SLE (0.86), its model size is notably smaller (5.70), indicating a more conservative approach that may exclude some relevant features. In contrast, aISSE strikes a better balance between sensitivity and precision, achieving the highest True Positive Rate (TPR = 0.98) among all methods, although with a slightly higher False Discovery Rate (FDR = 0.15). Despite the increased FDR, the method maintains a comparable model size, thus achieving strong predictive performance without sacrificing sparsity.

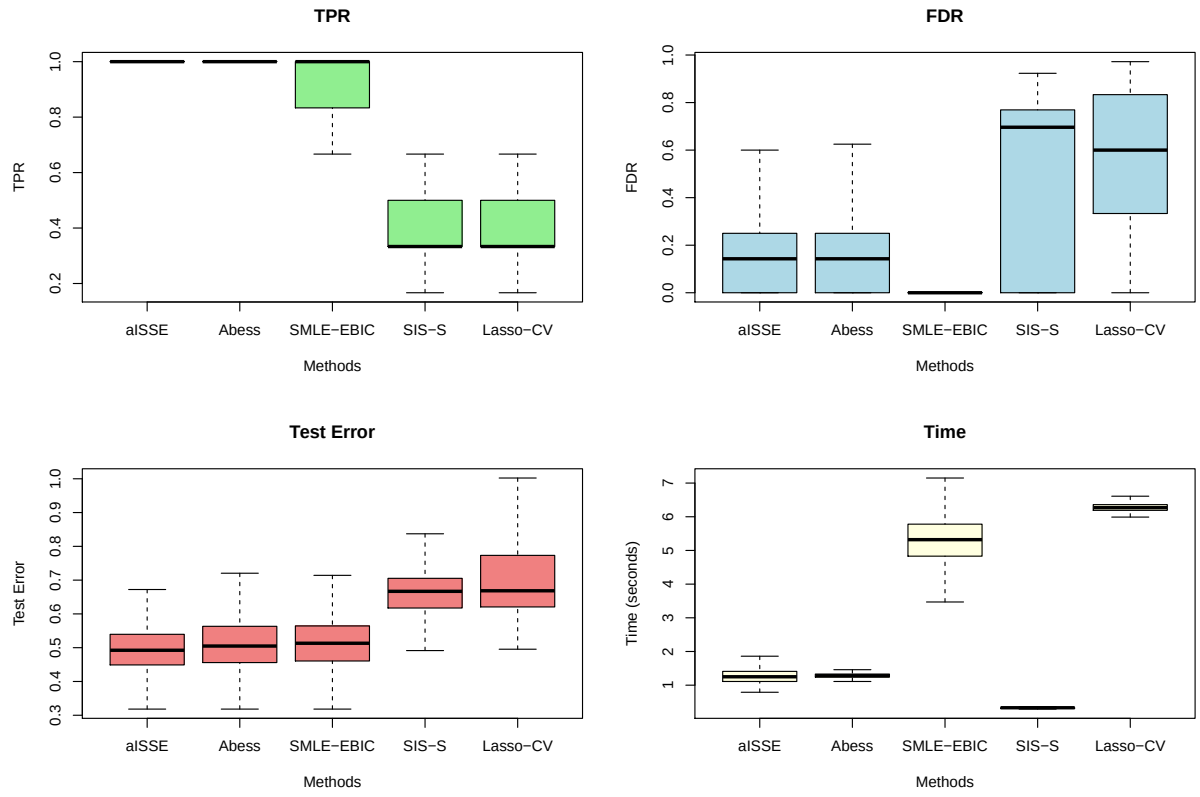


Figure 3.4: Box plots comparing the performance of five feature selection methods (aISSE, Abess Select, SMLE-EBIC, SIS-S, Lasso-CV) across four metrics: TPR, FDR, Test Error, and Time. Each box plot illustrates the distribution of the respective metric across 100 simulations.

The performance of these methods is compared visually in Figure 3.4, which shows side-by-side box plots of the performance metrics across 100 simulations for each method. The box plots clearly illustrate aISSE's better performance in TPR and Test Error. Overall, the aISSE method demonstrates robust performance across all metrics, particularly excelling in TPR and Test Error, making it the most reliable choice for scenarios where accurate feature recovery is critical.

3.5 Application to CLSA Metabolite Data

In this section, we applied feature selection to a metabolite dataset. Metabolites are molecules that are intermediate or end products of cellular metabolic pathways. These metabolites are extracted from biological samples, including fluids, cells, and tissues. In humans, there are hundreds of thousands of known metabolites [68]. Metabolomics research seeks to quantify metabolite levels and discover how metabolite profiles are related to both normal and pathological conditions, including diseases; see Fuller et al. (2023) [20] for a recent review of the role of metabolomics in epidemiology.

For this analysis, we used data available from the Comprehensive Cohort of the Canadian Longitudinal Study on Aging (CLSA) [47]. Briefly, from 2010 to 2015, the CLSA recruited participants between the ages of 45-85 with the aim of identifying biological, social and environmental factors related to aging and disease. The comprehensive cohort includes over 30,000 participants; however, metabolite data was only available on 8,229 of those individuals. Information about how the samples were processed and the metabolites were measured can be found in Michelotti et al. [38].

A total of 1,091 metabolites were available for this analysis. Each metabolite corresponds to a continuous feature in our dataset. The proportion of missing metabolite values ranges from 0 to 12%. However, 1061 metabolite features are missing for less than 5% of the participants. To handle missing data, we impute the missing value with the lowest observed value; this approach is recommended since missing metabolite values are most likely below detection limits [44]. To examine sensitivity of results to the imputation strategy, we also imputed with the median value.

The goal of this analysis was to assess the efficacy of various feature screening techniques in using metabolites to predict intraocular pressure (IOP), which is a modifiable risk factor for glaucoma. IOP is measured in both eyes, so the outcome of interest is the average of the two IOP measurements. Some participants had IOP measured on one eye only so only the one value was used. We used corneally-compensated IOP since it is adjusted for corneal properties. IOP is a continuous variable that ranges from 2.1 to 57.7 and has an average value of 16 in the CLSA dataset. We applied the same feature screening methods and performance metrics as described in Section 3.3.

3.5.1 Screening with a Fixed Model Size

In this scenario, the 8,229 participants were randomly partitioned into training and testing datasets, with the testing set accounting for 20 percent of the total or 1,646 partipants. Feature screening was performed on the training dataset and the top 9 features were retained with the exception of Lasso. For Lasso, we can only specify the maximum number of features so we set this to 10. The model size of 9 features was chosen to standardize the number of features retained between the different methods; when not constrained, Abess selects approximately 8 features.

These features were then utilized to predict IOP in the testing dataset and test error was computed. This process was repeated 100 times, with a different random training/testing set each time. The test error and computational time was saved for each testing set and the average was computed. For comparison purposes, we also include an approach where 9 features were randomly selected to be in the predictive model. This gives us a benchmark

for the worst-case test error.

The results of the first screening scenario are shown in Table 3.5. When model size is fixed at 9, all methods have very similar test error; values range from 15.43-15.70. On the other hand, randomly drawing 9 features results in an approximately 30% higher test error around 24. Computational time for both SMLE and ISSE is much higher than the other methods. For this particular scenario when the number of features to retain is predetermined, the increased computational time does not lead to observable improvements in test error.

Table 3.5: Comparison of average Model Sizes, Test Errors, and Times

Method	Model Size	Test Error	Time (seconds)
Random	9	23.23	0.00
SIS-S	9	15.50	0.49
Lasso-CV	7.29	15.49	0.08
SMLE-EBIC	9	15.70	2.54
Abess	9	15.43	0.12
ISSE	9	15.52	3.17

3.5.2 Selection

In this section, we do not assume a predefined model size. Instead, we perform feature selection using the various methods previously described but allowing each to flexibly select the best model size. This better mimics what is done during real data analyses since the model size is typically not known.

To mimic real-life practice, we used the default settings coded in the packages corresponding to each method. This is similar to the strategy used in Section 3.4. As with the

scenario with fixed model size, we again split the participants into testing and training datasets. The data splitting was repeated 100 times and the performance was assessed with average test error and computational time over the 100 data splits.

Results comparing the methods using default selection strategies are summarized in Table 3.6. It provides a comparative analysis of different methods using default selection strategies across median and minimum imputation approaches. The results highlight substantial differences in average model sizes, test errors, and computational times. For both imputation methods, SIS-S consistently selected the largest models, retaining over 600 features, while aISSE produced the smallest models with only 2 features, indicating a more parsimonious approach.

Test errors across the methods (except SIS-S) were relatively similar, ranging from 15.24 to 17.04, with minimal variation between the median and minimum imputation strategies, suggesting consistent predictive accuracy. Computational efficiency was also a key differentiator: aISSE demonstrated the fastest runtimes for both imputation techniques. In median imputation, aISSE was approximately 32.6% faster than Lasso-CV and significantly outperformed Abess and SMLE, with runtimes about 81% and 81.6% lower, respectively. For minimum imputation, aISSE maintained its lead in efficiency, running about 80% faster than Abess and 83.5% faster than SMLE.

Overall, the results demonstrate that aISSE offers a favorable balance of model simplicity, computational speed, and predictive accuracy, making it a robust choice across different imputation methods.

Table 3.6: Comparison of Average Model Sizes, Test Errors, and Times for Median and Minimum Imputation Methods

Median Imputation			
Method	Model Size	Test Error	Time (seconds)
aISSE	2.01	15.58	16.10
SIS-S	610.21	16.86	47.07
Lasso-CV	9.59	15.27	23.89
SMLE-EBIC	4.21	15.48	87.49
Abess	9.35	15.24	81.05
Minimum Imputation			
Method	Model Size	Test Error	Time (seconds)
aISSE	2.01	15.61	15.09
SIS-S	619.77	17.04	54.02
Lasso-CV	6.47	15.50	35.14
SMLE-EBIC	2.43	15.66	96.89
Abess	8.45	15.42	81.08

3.5.3 Feature selection path for a run of aISSE

In adaptive ISSE (aISSE), we iteratively expand the active set by incorporating potential features from the inactive set and reduce the model size using the extended Bayesian information criterion (EBIC). To examine the feature path, we consider results from a single split of the metabolite dataset. For this dataset, the selection process required 12 steps, with the model size decreasing from 140 to 2 over the 12 steps.

We refitted models for each of the 12 steps and recorded the coefficients for all potential causal metabolites at each step. Across the 12 steps, a total of 140 metabolites are included in the final models. We assume that the magnitude of more important metabolites will be

larger than the less relevant metabolites. Therefore, to determine the best 10 out of the 140 metabolites to plot, we ranked each metabolite by the sum of the magnitudes of the corresponding regression coefficients.

Figure 3.5 traces the coefficient values for the top 10 metabolites across the 12 steps. We note that coefficient values do change from step to step, with all but two moving to the inactive set by the last step. Notably, the two features selected in the final model, vanillylmandelate (VMA) and mannose, consistently remain in the model and their coefficients increased over the steps. This consistent inclusion lends weight to a true association between these metabolites and IOP

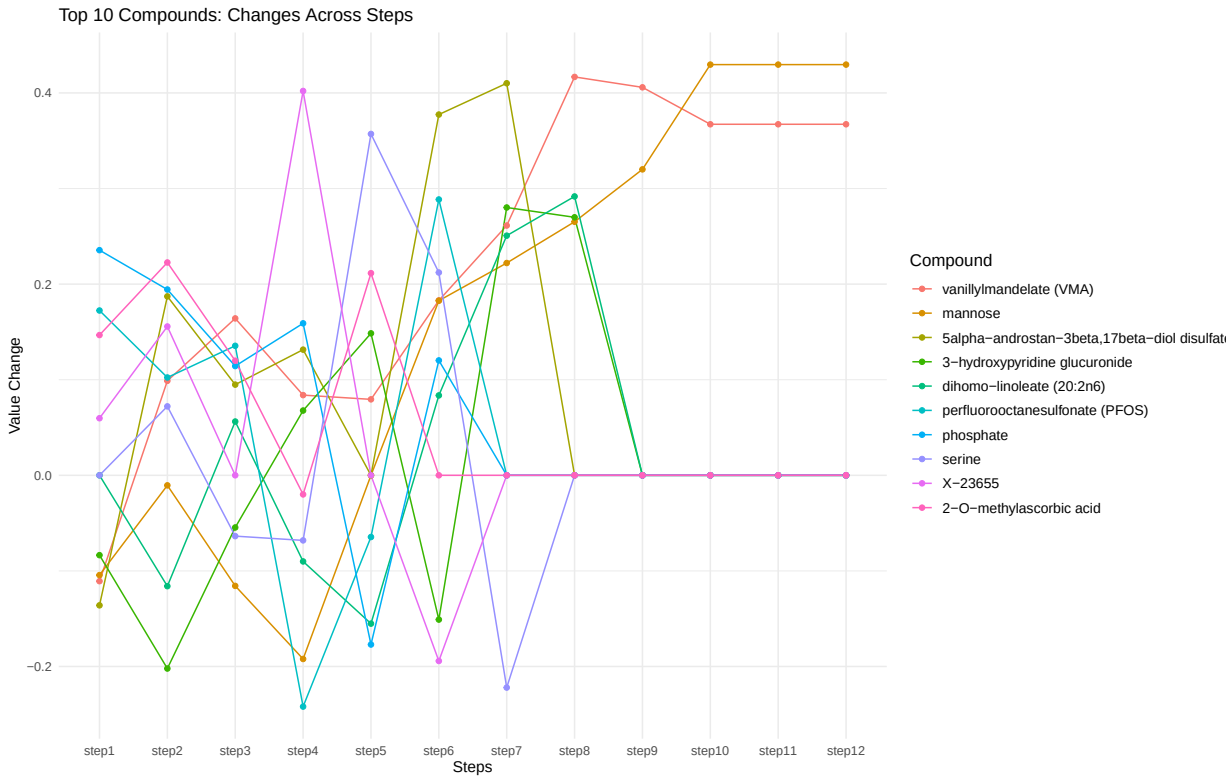


Figure 3.5: Coefficient paths of the selected metabolites during the adaptive ISSE (aISSE) feature selection process. The figure shows the magnitudes of the coefficients for each metabolite from step 1 to step 12.

3.5.4 Biological plausibility of the selected metabolites

Using aISSE, our study identified (VMA) and mannose as key metabolites associated with intraocular pressure (IOP). We now provide background information on these two metabolities in order to investigate whether the results are biologically plausible.

Vanillylmandelate (VMA)

VMA, a primary metabolite of catecholamines such as epinephrine and norepinephrine, plays a vital role in the sympathetic nervous system. Elevated levels of catecholamines have been linked to increased IOP, suggesting that VMA may serve as a biomarker for sympathetic nervous system activity affecting ocular pressure [41]. This relationship is supported by existing literature [21] indicating that stress and sympathetic nervous system overactivity can contribute to elevated IOP, which could further lead to conditions such as glaucoma.

Mannose

Mannose, a simple sugar and an essential component of glycoproteins and glycolipids, is involved in various metabolic pathways [49]. Its role in cellular communication and immune response may influence ocular health. Research has shown that mannose metabolism can affect glycosylation processes, which are critical for maintaining the structural integrity of ocular tissues. Abnormal glycosylation has been implicated in the pathogenesis of glaucoma [2], suggesting that mannose levels could be directly related to changes in IOP [36].

In conclusion, our study identified VMA and mannose as important metabolites in the context of IOP, suggesting their potential as biomarkers and therapeutic targets. These findings contribute to a growing body of evidence linking metabolic processes to ocular

health, offering promising directions for future research and clinical practice. Further studies is needed to determined if there relationship are replicated in other studies.

3.6 Discussion

In this Chapter, we introduced ISSE and adaptive ISSE, methods designed for efficient feature screening and feature selection in ultrahigh-dimensional datasets. Through rigorous testing on both simulated data and real-world metabolomics data, we have demonstrated ISSE's capability to handle complex dependencies among features while achieving high accuracy and computational efficiency. Additionally, we presented aISSE, an adaptive variant of ISSE, which can dynamically determine the model sparsity, further enhancing the selection process.

Looking ahead, we aim to extend ISSE's functionality to address a wider array of data analysis challenges and incorporate it into distributed computing environments to better accommodate the complexities of big data. Our ultimate goal is to establish ISSE and aISSE as versatile and robust tools that enable researchers and data scientists to make precise and swift decisions across various disciplines.

Chapter 4

BANS: Real-Time Streaming Feature Selection for Ultrahigh-Dimensional Data

4.1 Introduction

In modern scientific research, the volume and complexity of datasets are growing exponentially with new features. This influx leads to two significant challenges: firstly, the need to screen out irrelevant features, a necessity dictated by constrained storage and computational resources, and secondly, identification of causal features that may alter the characteristics of samples [35, 42, 48, 80, 81]. For example, in medical genetics, data from millions of variants are collected to identify those linked to specific diseases. In cybersecurity, millions of data packets are scanned in real-time to detect spam and viruses [24].

In the world of finance, predicting a company's future isn't just about traditional metrics anymore but involves constantly updating with fresh data, requiring a fast reaction to companies in danger of bankruptcy [28]. Similarly, social media platforms like X (Twitter) and Facebook identify real-time trends and hot topics through endless streams of content [72]. Streaming data, such as X or Facebook data, refers to data that is continuously generated by various sources. This type of data needs to be processed incrementally using stream processing techniques to derive insights and make real-time decisions.

Researchers have developed various approaches to select important features from streaming data. Perkins [43] proposed a method called "grafting". However, grafting is not truly real-time, as all features must be known before processing can begin. In 2005 and 2006, Zhou and co-authors introduced two methods: "Information-Investing" [83] and "Alpha-Investing" [82]. Alpha-Investing models the process as linear regression, computing the p-value for each feature. This method requires knowledge of some parameters, such as the initial wealth and decrease rate, which may need to be tuned for different datasets, while also assuming normally distributed errors. Additionally, Alpha-Investing evaluates features one at a time, resulting in increased running time as the number of features grows. OSFS [69] focuses on selecting features with weak (but non-redundant) or strong relevance, utilizing online relevance and redundancy analyses. Features with high marginal dependence are selected through relevance testing, but they are removed if they exhibit low conditional dependence with other features in the active set. SAOLA [77] emphasizes pairwise relationships between features, calculating them using mutual information for high-dimensional data. A drawback of both SAOLA and OSFS is that their estimation of joint effects among features is limited; they rely on conditional dependence and

mutual information, which can only measure pairwise interactions rather than simultaneously considering all features. To address these joint effects, Wang [65] introduced OGFS, a framework designed for online feature selection that leverages pre-existing knowledge about group structure and includes both intra-group and inter-group feature selection. Yu [78] extended this approach, adapting scalable and accurate online feature selection to group feature selection challenges in sparse scenarios. Liu et al. [33] proposed a framework for online multilabel group streaming feature selection, incorporating group structure analysis. However, the joint effects represented by intra-group dependence are often insufficient, and the information gain defined by these methods can be inefficient due to the necessity of accounting for all pairwise and three-way interactions, making the procedure impractical.

To address these limitations in estimating joint effects and improving the efficiency of feature selection, we introduce the Batch Adapted Neighbour Searching (BANS) method. BANS aims to approximate the sparsity-restricted maximum likelihood estimator (SMLE) [73] incrementally with each new feature. At each step, when new features are introduced, we perform SMLE estimation on the feature space, combining the active feature set from the previous step with the incoming features. Recognizing the limitations of SMLE in finding only local log-likelihood optima, BANS employs a search strategy that explores the neighborhood through various step sizes. This leads to consideration of several candidate models to locate the optimum. Unlike model-free methods such as OSFS and SAOLA, BANS incorporates a model assumption, enabling more efficient detection of joint effects. The performance of BANS is illustrated using both simulated and real datasets.

This chapter is structured as follows. Section 4.2 outlines the BANS methodology, offering a detailed discussion of the algorithmic procedures. Sections 4.3 and 4.4 present

extensive simulations and real-world experiments to demonstrate the benefits of BANS compared to existing feature screening techniques, particularly with datasets that exhibit complex correlation patterns. Lastly, Section 4.5 concludes with a summary of the main findings and proposes future research directions in the field of feature screening.

4.2 Methodology

4.2.1 Notation and Streaming feature screening setup

Assume that $\boldsymbol{X} = \{\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\} \mid \boldsymbol{x}_i = (x_{i1}, \ldots, x_{ip})\}$ represents the entire dataset, where $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)^T$ denotes the observed decision class of the n samples, which remains fixed for all timestamps.

According to the definition of feature streaming, only q features from $\boldsymbol{X}$ arrive at the machine at timestamp t. Let $\{\boldsymbol{v}_1^{(t)}, \boldsymbol{v}_2^{(t)}, \ldots, \boldsymbol{v}_q^{(t)}\}$ be the subset of $\boldsymbol{X}$ arriving at t, where $\boldsymbol{v}_j^{(t)} = (v_{1j}^{(t)}, \ldots, v_{nj}^{(t)})^T$ for $j = 1, \ldots, q$. Due to limited storage, the retained feature set at the t^{th} timestamp is $\{\boldsymbol{u}_1^{(t)}, \boldsymbol{u}_2^{(t)}, \ldots, \boldsymbol{u}_k^{(t)}\}$, where $\boldsymbol{u}_j^{(t)} = (u_{1j}^{(t)}, \ldots, u_{nj}^{(t)})^T$ for $j = 1, \ldots, k$.

Denote $\boldsymbol{U} \in \mathbf{R}^{n \times k}$ and $\boldsymbol{V} \in \mathbf{R}^{n \times q}$ as two matrices of features. The working feature matrix at time t is given by $\boldsymbol{X}^{(t)} = [\boldsymbol{U}^{(t)}, \boldsymbol{V}^{(t)}]$, where the i-th sample can be rewritten as

$$\boldsymbol{x}_i^{(t)} = (u_{i1}^{(t)}, u_{i2}^{(t)}, \ldots, u_{iq}^{(t)}, v_{i1}^{(t)}, v_{i2}^{(t)}, \ldots, v_{ik}^{(t)}).$$

Given the GLM model mentioned in Section 2.2, the log-likelihood function of $\boldsymbol{\beta} =$

$(\beta_1^{(t)}, \beta_2^{(t)}, \ldots, \beta_{q+k}^{(t)}) \in \mathbf{R}^{q+k}$ at timestamp t is given by

$$l(\boldsymbol{\beta}) = \sum_{i=1}^{n} [y_i \cdot \boldsymbol{x}_i^{(t)} \boldsymbol{\beta} - b(\boldsymbol{x}_i^{(t)} \boldsymbol{\beta})]. \tag{4.1}$$

Streaming feature screening aims to retain the causal features at each timestamp. Let the causal feature set be denoted by $s^* = \{j : \beta_j^* \neq 0, j = 1, \ldots, q+k\}$, where β^* represents the true coefficients of the features. The estimated sparse pattern is given by $\tilde{s} = \{j : \tilde{\beta}_j \neq 0, j = 1, \ldots, q+k\}$, where $\tilde{\beta}_j$ denotes the estimated coefficients of the features. Assuming a maximum storage size of k, the SMLE estimator at timestamp t is defined by

$$\boldsymbol{\beta}_{\text{SMLE}}{}^{(t)} = \arg\max_{\boldsymbol{\beta}} l(\boldsymbol{\beta}) \quad \text{subject to} \quad ||\boldsymbol{\beta}||_0 \leq k, \tag{4.2}$$

As shown in [73], with a $\boldsymbol{\gamma}$ close to $\boldsymbol{\beta}$, one can approximate $l(\boldsymbol{\beta})$ in (4.2) by a surrogate function

$$h(\boldsymbol{\beta}, \boldsymbol{\gamma}) = l(\boldsymbol{\gamma}) + (\boldsymbol{\beta} - \boldsymbol{\gamma})^\top l'(\boldsymbol{\gamma}) - (\tau/2)||\boldsymbol{\beta} - \boldsymbol{\gamma}||_2^2, \tag{4.3}$$

In the surrogate function $h(\boldsymbol{\beta}, \boldsymbol{\gamma})$, the parameter τ acts as a regularization term that controls the curvature of the approximation. Specifically, at timestamp t, given the estimated coefficients from the previous timestamp, denoted as $\widehat{\boldsymbol{\beta}}^{(t-1)} = (\widehat{\beta}_1^{(t-1)}, \widehat{\beta}_2^{(t-1)}, \ldots, \widehat{\beta}_k^{(t-1)})$, and assuming an initial value of zero for the new features, we define the initial coefficient at timestamp t as $\boldsymbol{\beta}_0^{(t)} = (\widehat{\boldsymbol{\beta}}^{(t-1)}; 0) \in \mathbb{R}^{q+k}$. With this initialization, we can seek a local approximation of (4.2) at the t_{th} timestamp by

$$\boldsymbol{\beta}_{\text{approx.}}^{(t)} = \arg\max_{\boldsymbol{\beta}} h(\boldsymbol{\beta}, \boldsymbol{\beta}_0^{(t)}) \quad \text{subject to} \quad ||\boldsymbol{\beta}||_0 < k. \tag{4.4}$$

Hence,

$$\min_{\boldsymbol{\beta}} \frac{1}{2} \left\| \boldsymbol{\beta} - \tau^{-1} \left[\tau \boldsymbol{\beta}_0^{(t)} + \boldsymbol{X}^{(t)^T} \left(\boldsymbol{y} - b'(\boldsymbol{X}^{(t)} \boldsymbol{\beta}_0^{(t)}) \right) \right] \right\|_2^2 \quad \text{subject to } \|\boldsymbol{\beta}\|_0 \leq k. \tag{4.5}$$

Substituting $\boldsymbol{X}^{(t)} = [\boldsymbol{U}^{(t)}, \boldsymbol{V}^{(t)}]$ and $\boldsymbol{\beta}_0^{(t)} = (\widehat{\boldsymbol{\beta}}^{(t-1)}; 0)$, equation 4.5 becomes:

$$\min_{\boldsymbol{\beta}} \frac{1}{2} \left\| \boldsymbol{\beta} - \tau^{-1} \left[\tau(\widehat{\boldsymbol{\beta}}^{(t-1)}; \mathbf{0}) + \begin{bmatrix} \boldsymbol{U}^{(t)^T} \\ \boldsymbol{V}^{(t)^T} \end{bmatrix} \left(\boldsymbol{y} - b' \left(\begin{bmatrix} \boldsymbol{U}^{(t)} & \boldsymbol{V}^{(t)} \end{bmatrix} (\widehat{\boldsymbol{\beta}}^{(t-1)}; 0) \right) \right) \right] \right\|_2^2 \tag{4.6}$$
$$\text{subject to } \|\boldsymbol{\beta}\|_0 \leq k.$$

If (4.6) did not have the sparsity constraint, the unconstrained solution $\tilde{\boldsymbol{\beta}}^{(t)}$ would be

$$\tilde{\boldsymbol{\beta}}^{(t)} = \widehat{\boldsymbol{\beta}}^{(t-1)} + \tau^{-1} \left[\boldsymbol{U}^{(t)^T}, \boldsymbol{V}^{(t)^T} \right] R^{(t-1)}, \tag{4.7}$$

where $R^{(t-1)} = \boldsymbol{y} - b' \left(\begin{bmatrix} \boldsymbol{U}^{(t)} & \boldsymbol{V}^{(t)} \end{bmatrix} (\widehat{\boldsymbol{\beta}}^{(t-1)}; 0) \right)$ is the residual from the previous step. Consequently, $\boldsymbol{\beta}_{\text{approx.}}$ in (4.4) has an analytical expression,

$$\widehat{\boldsymbol{\beta}}^{(t)} = H_k[\tilde{\boldsymbol{\beta}}^{(t)}], \tag{4.8}$$

where $H_k[\boldsymbol{\beta}]$ is the hard-thresholding operator setting all but the k largest components in $|\boldsymbol{\beta}|$ to zero.

It is important to note that $\tilde{\boldsymbol{\beta}}$ represents a local maximum of $l_n(\boldsymbol{\beta})$, and $\boldsymbol{V}^{(t)}$ is a subset of $\boldsymbol{X}$. In the streaming feature setting, ensuring that $s^* = \{j : \boldsymbol{\beta}_j^* \neq 0,\ j = 1, \ldots, q+k\}$ is included in the selected subset $\tilde{s}$ is challenging. To enhance the screening accuracy of the

SMLE, recall from Section 3.2:

$$\max\left\{\left|\tau^{-1}\cdot x_j^\top(y-b'(\mathbf{X}\tilde{\boldsymbol{\beta}}))\right| : j\notin\tilde{s}\right\} < \min\left\{\left|\tilde{\beta}_j\right| : j\in\tilde{s}\right\}.$$

We define $\tilde{\rho}_j = \mathbf{x}_j^\top(\mathbf{y}-b'(\mathbf{X}\tilde{\boldsymbol{\beta}}))$. Thus,$\tilde{\rho}_j$ measures the correlation between the regression residual and the features not included in the selected model $\tilde{s}$. When $\tilde{\boldsymbol{\beta}}$ is significant, some features $\mathbf{x}_j$ may still be excluded due to the small effect of τ^{-1}. As $\boldsymbol{\beta}^{(t)}$ converges to a locally optimal solution, the strategy is to increase the step size appropriately to search for potentially relevant features that may have been overlooked.

4.2.2 Batch Adapted Neighbor Searching Algorithm

We propose the Batch Adapted Neighbor Searching algorithm to address the problem discussed in Section 4.2.1. This algorithm adaptively selects the step size τ^{-1}, allowing for the effective joint selection of relevant features within the iterative hard thresholding framework.

Denote $|\tilde{\beta}_j|_{(h)}(\tilde{s})$ as the h-th order statistic (smallest) in the set $\{|\tilde{\beta}_j| : j\in\tilde{s}\}$. Let $\tilde{\rho}_j = \mathbf{x}_j^\top(\mathbf{y}-b'(\mathbf{X}\tilde{\boldsymbol{\beta}}))$, and denote $|\tilde{\rho}_j|_{(|\tilde{s}^c|-h)}(\tilde{s}^c)$ as the $|\tilde{s}^c|-h$-th order statistic in the set $\{|\tilde{\rho}_j| : j\in\tilde{s}^c\}$, which is equivalent to the h-th largest statistic in $\{|\tilde{\rho}_j| : j\in\tilde{s}^c\}$. Given a suitable $h<k$, the idea is to increase τ^{-1} to τ_h, which satisfies

$$\tau_h^{-1}||\tilde{\rho}_j|_{(|\tilde{s}^c|-h)}| = \frac{|\tilde{\beta}_j|_{[h]}(\tilde{s}) + |\tilde{\beta}_j|_{[h+1]}(\tilde{s})}{2}.$$

Next, we attempt to perform one more iteration based on $\tilde{\boldsymbol{\beta}}$ and obtain a new update

for $\boldsymbol{\beta}$, which can be expressed as

$$\tilde{\beta}_h = H\left(\tilde{\boldsymbol{\beta}} + \tau_h^{-1}\mathbf{X}^\top\left\{\mathbf{y} - b'(\mathbf{X}\tilde{\boldsymbol{\beta}})\right\}; k\right). \tag{4.9}$$

Intuitively, the update in (4.9) facilitates an exchange of features between the active set $\tilde{s}$ and the inactive set $\tilde{s}^c$. This happens because the top h-th largest elements are updated to be at least as large as the h-th smallest coefficient features in the active set $\tilde{s}$. After applying the hard thresholding operator H_k, these features are retained in the model if the likelihood increases.

This approach can be incorporated into the framework of the iterative hard thresholding algorithm by appropriately searching for a step size τ_h. By examining h different models, we can select the best model with the largest likelihood, similar to Equation 4.8:

$$\widehat{\boldsymbol{\beta}}^{(t)} = \arg\max_{\beta_h} L\left(H_k\left[\tilde{\boldsymbol{\beta}}_h^{(t)}\right]\right), \tag{4.10}$$

The procedure is detailed in Algorithm 4.1.

4.3 Simulation

4.3.1 Data Simulation and Method Implementation

To illustrate the functionality and performance of the BANS method, we conducted several simulation studies. We generated datasets composed of five sub-datasets, each containing 400 observations (rows) and 300 features, using the `Gen_Data` function from the **SMLE**

Algorithm 4.1 Batch Adapted Neighbor Searching Algorithm

Input: streaming data response $\boldsymbol{y}$, retained model size k, current step t, previous estimate $\widehat{\boldsymbol{\beta}}^{(\boldsymbol{t-1})}$, retained features $\boldsymbol{U}^{(t-1)}$, incoming features $\boldsymbol{V}^{(t)}$, residuals $R^{(t-1)}$, log-likelihood $l(\widehat{\boldsymbol{\beta}}^{(t-1)})$ and searching range h.

Repeat until stopping criteria are satisfied:

{

Input $\boldsymbol{\beta}^{(t-1)}$, $\boldsymbol{U}^{(t)}$, $\boldsymbol{V}^{(t)}$, $R^{(t-1)}$, $l(\widehat{\boldsymbol{\beta}}^{(t-1)})$,

$\tilde{\rho}_j = \mathbf{x}_j^\top \left(\mathbf{y} - b' \left([\mathbf{U}^{(t-1)}, \mathbf{V}^{(t)}] \tilde{\boldsymbol{\beta}}^{(t-1)} \right) \right)$

Rank $|\tilde{\rho}_j|$ and Calculate $|\tilde{\rho}_j|_{[|\tilde{s}^c|-h]}$

Calculate $\tau_h^{-1} = \frac{|\tilde{\beta}_j|_{[h]}(\tilde{s}) + |\tilde{\beta}_j|_{[h+1]}(\tilde{s})}{2|\tilde{\rho}_j|_{[|\tilde{s}^c|-h]}}$

Update $\tilde{\beta}_h^{-1} = H_k(\boldsymbol{\beta}^{t-1} + \tau_h [\boldsymbol{U}^{(t-1)^T} R^{(t-1)}; \boldsymbol{V}^{(t)}(\boldsymbol{y} - b'(\mathbf{0}))])$

Select the step size $\tau^{-1} = \underset{\tau \in \{\tau_h | h=0,1,\ldots,s\}}{\arg\max} \log\Big(L(\tilde{\beta}_h) \Big)$

Update the model by $\widehat{\boldsymbol{\beta}}^{(\boldsymbol{t})} = H_k(\boldsymbol{\beta}^{t-1} + \tau^{-1} [\boldsymbol{U}^{(t-1)^T} R^{(t-1)}; \boldsymbol{V}^{(t)}(\boldsymbol{y} - b'(\mathbf{0}))])$

Update residual $R^{(t)} = \boldsymbol{y} - b'([\boldsymbol{U}^{(t-1)}, \boldsymbol{V}^{(t)}] \widehat{\boldsymbol{\beta}}^{(t)})$

Output: $\widehat{\boldsymbol{\beta}}^{(\boldsymbol{t})}$, $\boldsymbol{U}^{(t)}$, $R^{(t)}$, $l(\widehat{\boldsymbol{\beta}}^{(t)})$

$t = t + 1$

}

package [79]. In each sub-dataset, only 5 features were causally linked to the response, which we refer to as the "key predictors". The remaining features did not influence the response and served as noise. These sub-datasets were then merged to form a "full dataset" with 1,500 features and 400 observations, resulting in a dataset with 25 causal features and streaming data simulated by the five sub-datasets.

Scenario	Response	Position in Feature Matrix	Correlation Structure
1	Gaussian	1, 2, 3, 4, 5	Independent
2		Random	Auto Regressive
3		1, 3, 5, 7, 9	Auto Regressive
4		1, 3, 5, 7, 9	Compound Symmetry
5	Binomial	1, 2, 3, 4, 5	Independent
6		Random	Auto Regressive
7		1, 3, 5, 7, 9	Auto Regressive
8		1, 3, 5, 7, 9	Compound Symmetry

Table 4.1: Summary of Simulation Datasets

We evaluated the BANS method across eight scenarios, summarized in Table 4.1. In Scenarios 1 to 4, the response follows a Gaussian distribution, while in Scenarios 5 to 8, it follows a Binomial distribution. Scenario 1 establishes a baseline with all features mutually independent, serving as a simple correlation benchmark for performance evaluation. In Scenario 2, features exhibit auto-correlation with $\rho = 0.8$, and causal features are randomly distributed.

Scenario 3 introduces a structured placement of causal features in specific positions within each sub-matrix, as defined in Equation 4.11. Specifically, the 1st, 3rd, 5th, 7th, and 9th features of each sub-matrix were given effect sizes $(4, -4, 3, -5, 4)$ leading to the following linear model:

$$\mu = \sum_{i=1}^{5} 3x_{i1} - 5x_{i3} + 2x_{i5} - 4x_{i7} + 2x_{i9} \tag{4.11}$$

In the Gaussian Scenario 3, the response is sampled from a normal distribution, $N(\mu, \sigma^2)$.

Scenario 4 employs a jointly normal distribution of features within a compound symmetry structure. In this setup, pairs of relevant features share a correlation of $\rho/2$, while all other feature pairs have a covariance of ρ.

For the Binomial scenarios, the response is sampled from a Binomial$(1, \pi)$ distribution, where

$$\log \frac{\pi}{1-\pi} = \sum_{i=1}^{5} \beta_{i1} x_{i1} - \beta_{i2} x_{i3} + \beta_{i3} x_{i5} - \beta_{i4} x_{i7} + \beta_{i5} x_{i9} \tag{4.12}$$

Here, the log-odds of π are defined as a linear combination of selected features with coefficients $\beta_{i1}, \beta_{i2}, \beta_{i3}, \beta_{i4}$, and β_{i5}, emphasizing the effect of each causal feature on the response.

Performance is assessed using several metrics. For Scenarios 1 to 4, where the response variable is continuous, performance is quantified using the average l_2 norm, which calculates the mean of the squared differences between the predicted and actual values. For Scenarios 5 to 8, where the response is binomial, performance is measured using log loss:

$$\text{Log Loss} = -\frac{1}{N} \sum_{i=1}^{N} \left[y_i \log(p_i) + (1 - y_i) \log(1 - p_i) \right]. \tag{4.13}$$

The log loss function evaluates the accuracy of the predictions in terms of the probability – a common approach when models predict probabilistic outcomes.

The Bayesian Information Criterion (BIC) is used to balance model fit against complexity, with lower BIC scores indicating more optimal models. The Positive Screening Rate (PSR) quantifies the method's effectiveness in identifying true causal features, with a higher PSR indicating a greater proportion of relevant features selected.

We compared BANS to several related methods. To ensure a fair comparison, each method was implemented to support batch processing of features. Specifically, a version of the Alpha-investing algorithm was coded in R, while for SIS, we utilized the open-source implementation available in the CRAN package **SIS** [53]. The implementations of OSFS, SAOLA, and FI (Feature Interaction [84]) were re-coded in R based on the MATLAB library **OSFS** [76], ensuring that all methods were evaluated under similar conditions. This approach allowed computation time to be compared based on the algorithm's performance, rather than the efficiencies of the coding language. OSFS and SAOLA used the Fisher Z test as a criterion, while FI employed conditional dependence, following their default settings.

4.3.2 Results for a single simulated dataset

To better understand performance, we first focus on the screening trajectory from a single dataset selected from Scenario 3 and Scenario 6. Figures 4.1(a) and 4.2(a) show the computation time versus timestamp. BANS and SAOLA have significantly lower computational times compared to all other methods. Figures 4.1(b) and 4.2(b) display PSR as a function of timestamp, with stars at the bottom of the plot indicating when causal features enter the data stream. Initially, all methods perform comparably; however, as more causal features are introduced, BANS outperformed the other approaches. From Figures 4.1(c) and 4.2(c), we observe that BANS has the largest training loss, but notably, its training loss does not increase as the algorithm progresses. This stability is essential as it indicates that BANSs learning curve remains steady without regressing or losing accuracy as more iterations are completed. In Figures 4.1 and 4.2(d), we see that all methods initially have similar test losses; however, as more features become available, the testing loss decreases,

and BANS ultimately achieves the lowest loss.

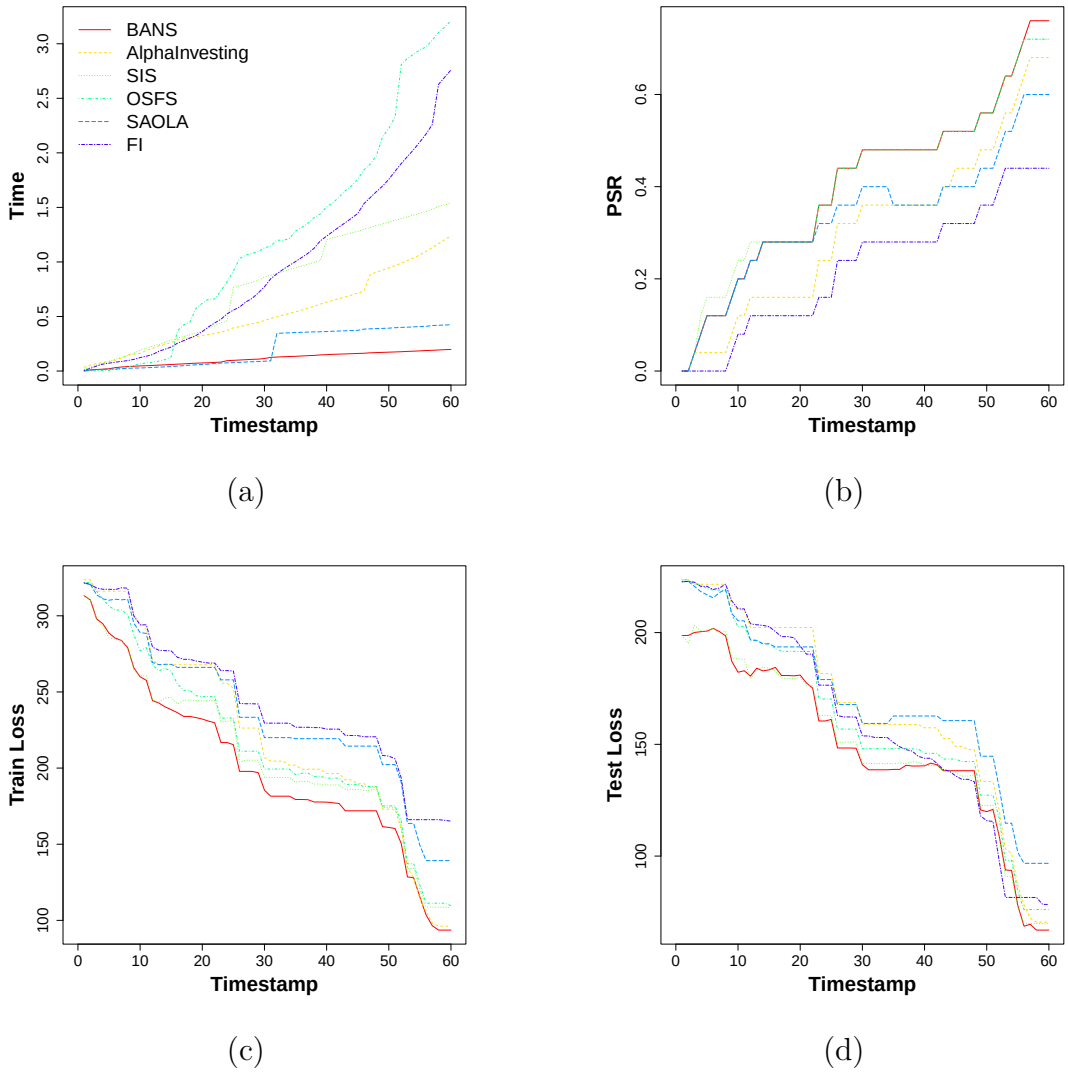


Figure 4.1: Comparison of performance under scenario 3. (a) Computation time vs timestamp. (b) PSR for each method at every timestamp. (c) Displaying the training loss as a function of timestamp. (d) Illustrating the test loss associated with these methods over timestamp.

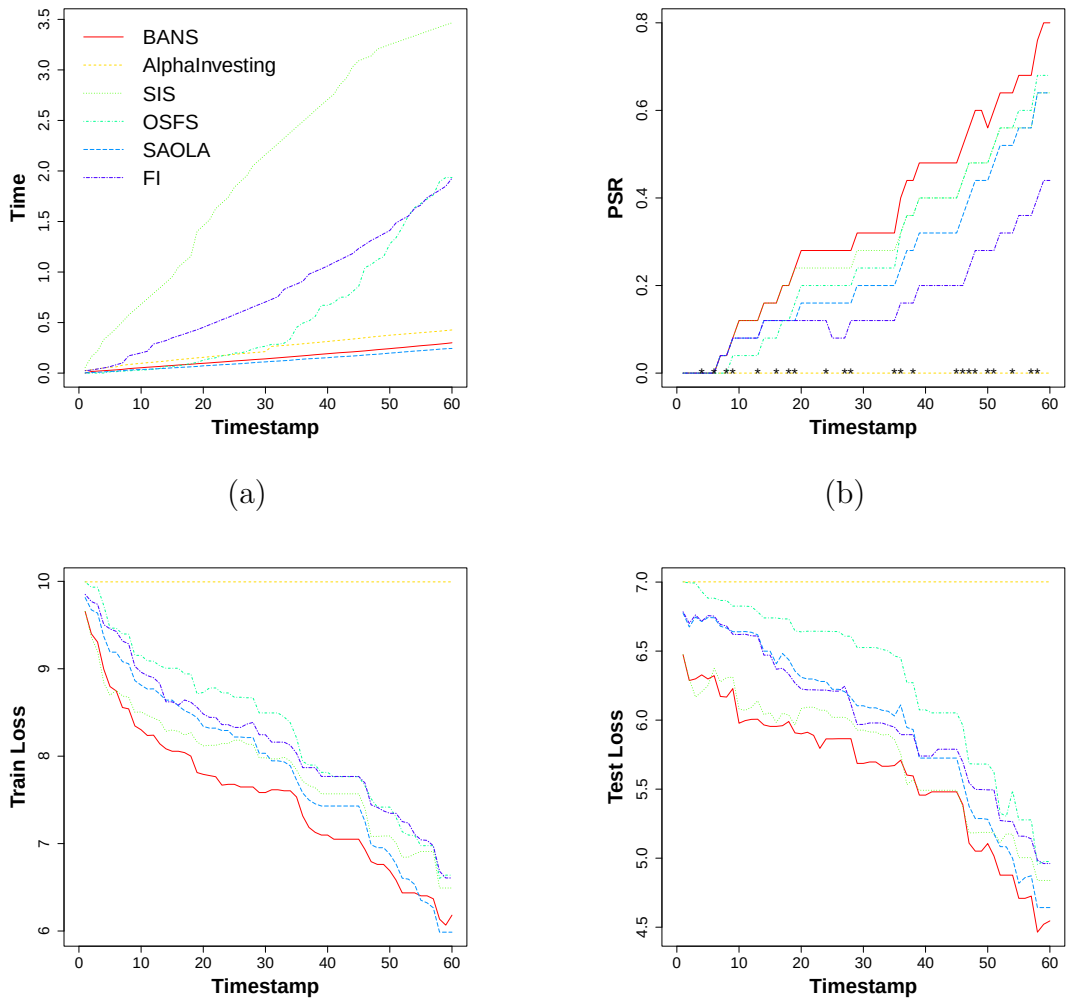
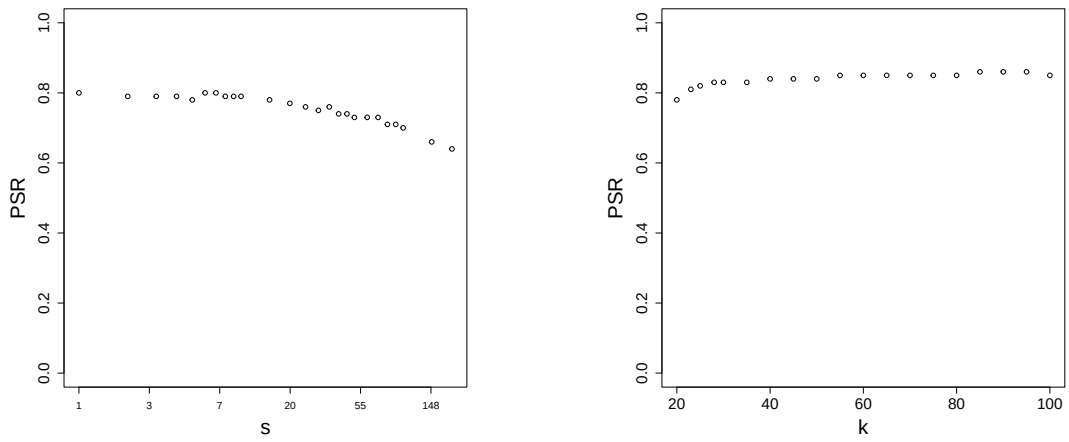


Figure 4.2: Comparison of performance under scenario 6. (a) Computation time vs timestamp. (b) PSR for each method at every timestamp. (c) Training loss as a function of timestamp. (d) Test loss as a function of timestamp.

4.3.3 Sensitivity analysis for BANS

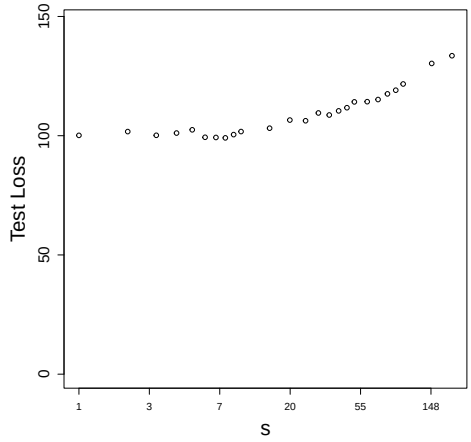
In this section, we evaluate the sensitivity of BANS to the size of the feature set arriving at each time stamp and to the number of features retained. The sensitivity tests are all conducted under Scenario 3, as described in Section 4.2, with 100 simulated datasets. Here, s denotes the number of features in each batch at every timestamp, while k represents the retained model size. The results are shown in Figure 4.3. For the plots versus s, we plot s on a log scale to better visualize performance at smaller s values. Without the log scale, the plot is dominated by higher s values.

Figure 4.3 demonstrates that BANS' performance is robust to variations in s. When s is set within an optimal range from 1 to 20, the PSR remains stable, indicating that BANS is capable of stable feature screening. When s becomes larger, for example 148, the PSR declines and test loss increases. This is due to a low number of iterations, as too many features in a batch limit the algorithm's available iterations. In contrast, the choice of k, which is not represented on a logarithmic scale, shows that as long as the number of retained features is above the number of causal features, the PSR and test loss remain stable. In fact, a slight increase in k over the true number of causal features can lead to a marginal improvement in PSR, suggesting that BANS can effectively manage a larger feature set without compromising accuracy. However, an excessively high k risks overfitting, as evidenced by an increased test error, underscoring the need for a balanced k to prevent deterioration in predictive performance. Together, these observations highlight that BANS is robust to small or moderate s and k, but that selecting extremes could negatively impact its feature selection and predictive accuracy.

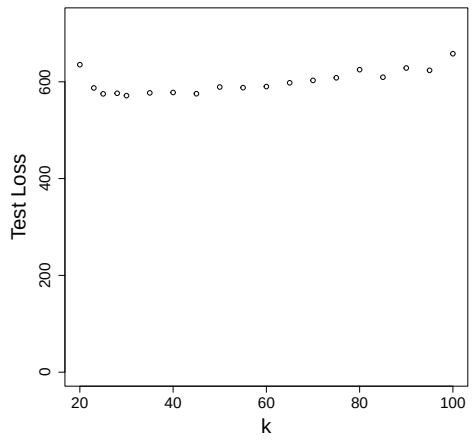


(a)

(b)



(c)



(d)

Figure 4.3: Sensitivity of BANS to size of feature sets and model size. (a) PSR versus the number of features that arrive at each timestamp (s). The x-axis is plotted on the log-scale to better visualize performance at smaller s values. (b) PSR as a function of model size (k). (c) Test loss as a function of number of features that arrive at each time stamp. The x-axis is plotted on the log-scale to better visualize performance at smaller s values. (d) Test loss as a function of model size (k).

4.3.4 Performance Evaluation Across Varied Scenarios

Results from scenarios 1 - 4 across 100 simulations are shown in Table 4.2. All datasets simulated for these scenarios included 400 samples and 1500 features, with a batch size of 25. The model size was fixed at 25 for BANS and SIS. BANS reliably demonstrated superior performance in terms of model selection and predictive accuracy, as indicated by its low BIC scores, high PSR values, minimal test and training losses, and fastest processing times. BANS exhibited robustness across various settings, including different correlation structures and causal feature placements. Its superiority was especially notable in the CS setting (Scenario 4), where it outperformed all other methods. Although the PSR estimated for BANS was within simulation error of the PSR of other methods, BANS reliably ranked at the top, even at much lower model sizes.

Table 4.2: Results from 100 Simulations Across Scenarios 1 - 4

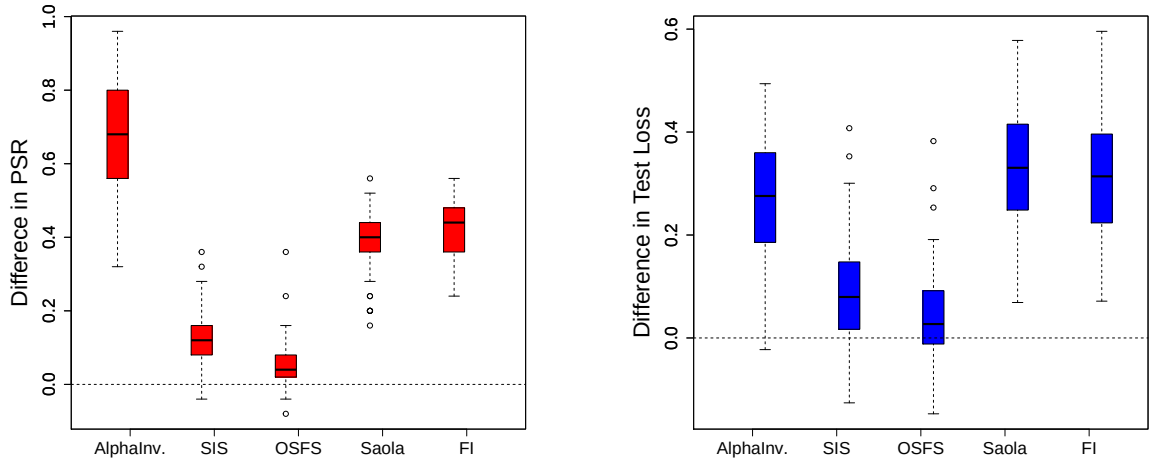
Methods	Model Size	PSR	BIC	Test Loss	Train Loss	Time
Scenario 1						
BANS	25.00	0.95	2354.85	22.32	17.77	0.13
AlphaInvesting	26.78	0.90	2578.28	39.07	30.66	0.84
SIS	25.00	0.80	2891.91	75.82	57.29	0.78
OSFS	23.27	0.91	2536.59	34.21	27.28	2.29
SAOLA	45.17	0.87	2763.39	53.00	34.24	0.26
FI	48.12	0.44	3426.95	261.24	146.27	1.67
Scenario 2						
BANS	25.00	0.95	2325.89	20.35	16.29	0.13
AlphaInvesting	28.42	0.90	2553.10	38.99	28.86	0.89
SIS	25.00	0.79	2927.40	81.51	60.54	0.76
OSFS	23.31	0.91	2523.13	33.13	25.87	2.44
SAOLA	43.66	0.87	2783.67	54.76	35.17	0.25
FI	49.63	0.45	3426.83	262.39	142.88	1.69
Scenario 3						
BANS	25.00	0.83	2735.58	54.06	38.95	0.13
AlphaInvesting	23.79	0.79	2835.36	74.56	56.81	0.75
SIS	25.00	0.68	3046.66	111.68	80.54	0.76
OSFS	21.10	0.77	2864.29	73.91	55.46	1.96
SAOLA	39.35	0.51	3273.20	209.52	113.57	0.25
FI	45.51	0.40	3389.71	262.87	138.29	1.58
Scenario 4						
BANS	25.00	0.93	2442.44	25.40	20.60	0.13
AlphaInvesting	40.46	0.88	2660.55	48.04	30.19	1.14
SIS	25.00	0.76	2968.75	91.00	69.17	0.82
OSFS	22.75	0.89	2634.28	41.07	32.90	1.94
SAOLA	18.42	0.73	3013.49	105.71	84.34	0.13
FI	28.31	0.45	3406.51	218.62	187.03	0.91

Table 4.3 displays the results for the simulations with a binomial response variable (scenarios 5-8). For these scenarios, each dataset had a sample size of 600 and a total of 1500 features, with a batch size of 25. The larger sample size was chosen since logistic

Table 4.3: Results from 100 Simulations Across Scenarios 5 - 8.

Methods	Model Size	PSR	BIC	Test Loss	Train Loss	Time
Scenario 5						
BANS	28.00	0.91	330.10	0.32	0.12	0.37
AlphaInvesting	8.93	0.26	704.72	0.60	0.53	0.56
SIS	28.00	0.79	443.08	0.35	0.21	3.18
OSFS	28.50	0.89	363.28	0.33	0.14	6.12
SAOLA	47.27	0.85	464.33	0.55	0.14	0.33
FI	41.75	0.45	698.70	0.61	0.36	1.72
Scenario 6						
BANS	28.00	0.90	407.74	0.34	0.19	0.37
AlphaInvesting	7.73	0.23	731.11	0.62	0.56	0.62
SIS	28.00	0.79	485.79	0.39	0.25	3.18
OSFS	28.50	0.87	426.02	0.36	0.20	5.55
SAOLA	47.27	0.85	522.65	0.47	0.18	0.32
FI	41.75	0.44	716.63	0.64	0.37	1.79
Scenario 7						
BANS	28.00	0.91	330.28	0.34	0.12	0.38
AlphaInvesting	8.85	0.26	703.64	0.60	0.53	0.62
SIS	28.00	0.79	442.26	0.35	0.21	3.04
OSFS	29.07	0.89	356.24	0.30	0.14	6.18
SAOLA	45.66	0.85	463.77	0.60	0.14	0.37
FI	41.58	0.45	698.10	0.62	0.35	1.95
Scenario 8						
BANS	28.00	0.81	436.94	0.40	0.21	0.35
AlphaInvesting	5.31	0.14	777.53	0.66	0.61	0.54
SIS	28.00	0.68	543.99	0.47	0.30	3.09
OSFS	26.76	0.76	471.63	0.43	0.25	4.65
SAOLA	39.83	0.43	723.42	0.72	0.39	0.29
FI	39.51	0.39	765.70	0.70	0.42	1.73

regression requires larger samples than linear regression. The model size was fixed at 28 for BANS and SIS. Compared to Table 4.2, the results in Table 4.3 show that all methods perform slightly worse in terms of PSR under the logistic model than the linear model. Despite this, BANS again outperforms all other methods. Although the computation time of BANS is similar to that of SAOLA, its PSR is always higher, and its test loss is substantially lower.



(a) (b)

Figure 4.4: Boxplots comparing performance across all methods for 100 simulated datasets under scenario 8. (a) Boxplot of the difference in PSR between BANS and each of the other methods. (b) Boxplot of the negative difference in test loss between BANS and each of the other methods.

The boxplot in Figure 4.4 provides a visual comparison of the performance of BANS against other methods under Scenario 8. In Figure 4.4(a), the y-axis represents the PSR difference between BANS and each of the other methods. Although the average PSR of BANS might not appear significantly higher at first glance, the boxplot reveals that in

the majority of simulations, BANS outperforms the other methods, with the first quartile positioned above zero for all comparisons. Instances where BANS has a lower PSR than other methods are infrequent. In Figure 4.4(b), the y-axis depicts the negative difference in test loss between BANS and each of the other methods. While OSFS and SIS are closer to BANS in their test loss values, approximately 75% of datasets have lower test loss with BANS than with either SIS or OSFS. In addition, Table 4.2 demonstrates that BANS achieves this performance in less than one-tenth of the computational time required by OFSF and SIS. This visualization highlights BANS's superiority, illustrating not only its strong average performance but also its consistency across a wide range of simulations. These results suggest that BANS is not only effective but also reliable, even when confronted with data variations that could influence model selection and feature identification.

4.3.5 Effect of very high-dimensional dataset

To determine how performance is affected by increased dataset dimensions, we simulated datasets with 8000 features. We maintained the same CS correlation framework as in Scenarios 4 and 8 but significantly increased the dimensionality of the feature space to 8000. To further challenge the methods, we reduced the effect sizes for the causal features and introduced weaker coefficients to the features, further complicating the task by 20%. The linear predictor is shown in equation 4.14. The response variables were simulated from $N(\eta_i, \sigma^2)$ (Scenario 4) and $\text{bin}(1, \pi)$ where $\log \frac{\pi}{1-\pi} = \eta$ (Scenario 8).

$$\eta = \sum_{i=1}^{5} (4x_{i1} - 5x_{i3} + 2x_{i5} - 4x_{i7} + 2x_{i9})/i \tag{4.14}$$

Table 4.4: Performance for simulation with very high number of features

Model	Methods	PSR	BIC	Test Loss	Time
Linear	BANS	0.87	6065.81	8.13	3.61
	AlphaInvesting	0.85	6209.73	9.22	28.74
	SIS	0.79	6471.25	11.27	5.88
	OSFS	0.84	6031.81	8.42	32.68
	SAOLA	0.70	6929.18	18.52	1.08
Logistic	BANS	0.69	1112.47	0.33	6.29
	AlphaInvesting	0.50	1342.40	0.33	10.01
	SIS	0.59	1349.25	0.32	47.05
	OSFS	0.65	1217.22	0.34	1280.59
	SAOLA	0.36	2049.44	0.54	5.47

Table 4.4 presents the performance of various methods under simulations with a very high number of features, evaluated on both linear and logistic models. For the linear model, BANS demonstrates the highest PSR at 0.87 and achieves a low test loss of 8.13, while also maintaining a relatively low computational time of 3.61. In comparison, AlphaInvesting achieves a similar PSR of 0.85, but requires significantly more computational time (28.74). Other methods such as SIS, OSFS, and SAOLA show lower PSR values and higher test losses, with SAOLA performing the worst with a PSR of 0.70 and a test loss of 18.52, though it is computationally the quickest. For the logistic model, BANS again outperforms other methods with a PSR of 0.69, a low test loss of 0.33, and a moderate computational time of 6.29. While OSFS achieves a slightly higher PSR (0.65), it requires substantially more time (1280.59). Across both models, BANS not only achieves a high PSR and low test loss but also does so with comparatively efficient use of computational time, indicating that BANS is more effective and efficient than the other methods tested in handling high-dimensional data.

4.4 Analyses based on real dataset

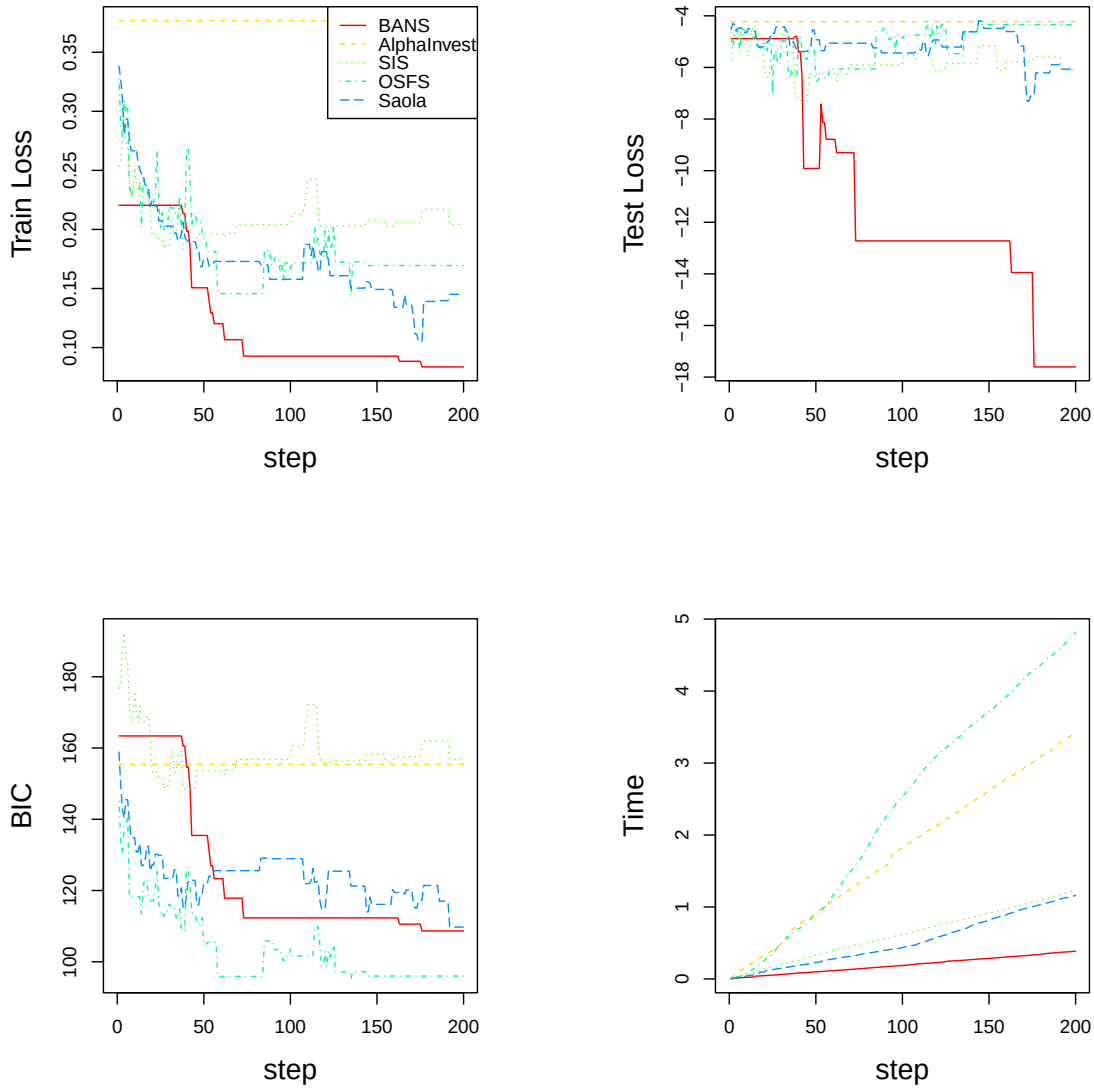
4.4.1 SMK-CAN-187 data

Technological advancements have enabled the collection of microarray data measuring gene expression for use in genetics, medicine, and diagnostics. Microarray datasets are typically high-dimensional and often include superfluous features that may not be relevant for classification tasks. An example of such a dataset is the SMK-CAN-187 [55], which contains gene expression data from both smokers diagnosed with lung cancer and those without the disease. The purpose of the dataset is to identify gene expression profiles that discriminate between the two groups. The dataset contains 19,993 gene expression features measured across 187 participants. We ran all methods on the SMK-CAN-187 dataset with features included in batches of size s = 100 to mimic streaming data. To assess performance using test or training loss, the dataset was split into a training set of size 120 and a test set size of 67.

Figure 4.5 shows the training and test loss, BIC and computational time across timestamps for the different methods. The application of these methods to the SMK-CAN-187 dataset reveals that BANS reliably maintains or does not decrease training loss. Importantly, it demonstrates a significant reduction in test loss, concurrently achieving the highest classification performance with shows in test loss among all methods evaluated.

Table 4.5 summarizes the performance using the metrics Bayesian Information Criterion (BIC), Model Size, Test Loss, and Time. We also consider the Relative Efficiency Score (RES), which is calculated by dividing the method's test loss and time by the minimum

Figure 4.5: SMK_CAN dataset learning path with model size 15 and feature intensity 100



values of these metrics across all methods and then multiplying the two quotients.(see equation 4.15)

$$\text{RES} = \frac{\text{test loss}}{\text{min text loss}} \frac{\text{time}}{\text{min time}} \tag{4.15}$$

A RES of 1 indicates the method has the lowest test loss and running time, denoting optimal efficiency and effectiveness. In this comparison, BANS, SIS, AlphaInvesting, OSFS, and SAOLA are also evaluated. For BANS and SIS, two model sizes were evaluated, 6 and 15.

Table 4.5: Performance of Different Methods on the SMK-CAN-187 Dataset

Methods	**BIC**	**Model Size**	**Test Loss**	**Time (s)**	**RES**
BANS	167.26	15.00	1.18	0.16	2.10
	153.15	6.00	0.76	0.13	**1.10**
SIS	199.01	15.00	0.74	0.70	5.77
	176.20	6.00	0.69	0.73	5.61
AlphaInvesting	127.49	7.68	1.06	6.73	79.41
OSFS	126.69	0.68	0.71	3.53	27.95
SAOLA	160.50	11.76	0.77	1.18	10.13

As presented in Table 4.5, the BANS method demonstrates both efficiency and effectiveness on the SMK-CAN-187 dataset. In particular, BANS achieves a comparable Test Loss to other methods with a much lower computational time, as shown by its RES value of 1.10, indicating a more efficient performance relative to alternatives. While methods such as SIS and SAOLA achieve similar or slightly lower Test Loss values, they require more time and have higher RES values, emphasizing the competitive advantage of BANS in handling this dataset efficiently.

4.4.2 Synthetic genetic dataset

To illustrate a second application of BANS, we applied it to a synthetic genetic dataset. Genomic association study datasets often contain an exceedingly large number of features, necessitating feature screening or selection methods in simulations. Due to the very large number of features, typically users must reduce the number of features before they can even use methods such as Lasso or SMLE for further model screening or selection. With BANS, however, we can create blocks of features and add them incrementally, as if they were streaming data. In this example, we evaluate this approach using a simulated genetic dataset.

In our experiment, we applied the BANS methodology to a simulated dataset in the genetic domain, aiming to identify associations between single-nucleotide polymorphisms (SNPs) and a response variable. To closely approximate real-world genetic complexity, we used genotype frequencies from the first phase of the 1000 Genomes Project, sampling through the **R** package **sim1000G**. This selection of SNP distributions reflects the genetic diversity within human populations, ensuring that the resulting dataset captures the complex correlation structures typical of genetic data. We particularly anticipated substantial non-linear correlations among SNPs located close to each other on the same chromosome, representing underlying biological processes.

The synthetic dataset contains a total of 10,031 SNPs, selected from chromosomes 14 to 22, based on genotypic data from 800 individuals. Each genotype is numerically encoded as 0, 1, or 2 to represent the count of minor alleles – the less frequent alleles within the sampled population. The response variable is continuous and assumed to be Gaussian

distributed. The screening size was set to 6, corresponding to the 6 causal features with large coefficient effects. For AlphaInvesting and SAOLA, model size control is not possible as they perform screening based on their default settings. To evaluate the sensitivity to feature order, we repeated the analysis 100 times, shuffling the input batch order each time.

Table 4.6: Simulation Results of Different Methods on the synSNPs Dataset Across 100 Repetitions

Method	**Model Size**	**BIC**	**Test Loss**	**Time (s)**	**Train Loss**	**RES**
BANS	6.00	3456.42	76.21	0.08	70.88	1.00
AlphaInvesting	51.04	3583.24	99.36	26.85	51.96	437.58
SIS	6.00	3418.87	77.59	0.45	65.57	5.73
OSFS	6.00	3454.36	78.44	9.71	70.57	124.93
SAOLA	24.66	3393.33	87.57	1.29	48.92	18.53

As shown in Table 4.6, while BANS achieves a similar Test Loss to the SIS and OSFS methods, it requires much less computational time than these methods. This efficiency is highlighted by its RES value of 1, indicating that BANS is more effective and efficient in handling high-dimensional data compared to the other methods tested.

4.5 Conclusion

In this chapter, we introduced BANS, a method tailored for the rapid and efficient selection of features in ultrahigh-dimensional datasets, including stream datasets. By testing BANS against simulation data and realistically structured data, we have demonstrated its ability to handle complex dependencies among features while maintaining high performance at much lower computation costs than other methods.

Although BANS currently relies on a predefined parameter k for feature retention, future work includes developing a strategy for the method to adaptively determine the best k. This improvement will further streamline the feature selection process and enhance the method's applicability.

Looking ahead, we plan to expand BANS' capabilities to address a broader range of data analysis challenges and integrate it into distributed systems to better manage the demands of big data. Our goal is to make BANS a more versatile tool that can assist researchers and data scientists in various fields to make informed decisions quickly and accurately.

Chapter 5

Conclusion and Future Work

This dissertation presents significant advancements to the analysis of ultrahigh and high dimensional data through the development and application of novel methods and software.

In Chapter 2, the R package **SMLE** was introduced. **SMLE** is the software implementation of the SMLE algorithm [73] for joint feature screening in ultrahigh-dimensional generalized linear models. By incorporating joint effects among features and providing flexibility with various screening parameters, SMLE is a computationally convenient and effective method. Through extensive simulations, we show that feature screening using **SMLE** yields more reliable screening results compared to existing approaches. Additionally, **SMLE** enables users to perform accurate post-screening feature selection through an accelerated Iterative Hard Thresholding procedure, complemented by the users preferred selection criteria, such as EBIC [7]. Visualization tools were also included in the R package, allowing users to easily interpret screening and selection results for inference or prediction purposes. **SMLE** can be downloaded from CRAN at https://CRAN.R-project.org/package=SMLE;

it has already been downloaded more than 35,000 times.

In Chapter 3, we introduced ISSE, a method designed for efficient feature selection and screening in the ultrahigh-dimensional data context. While SMLE uses a local search strategy, ISSE improves upon SMLE by incorporating splicing techniques to more effectively select relevant features. This approach ensures a stable solution within a finite number of steps and includes a specially designed information criterion that reliably determines the true sparsity level. Through rigorous testing on both simulated data and real-world metabolomics data, ISSE is shown to better handle complex dependencies among features than other approaches while achieving high accuracy and computational efficiency. Additionally, the adaptive variant aISSE dynamically determines model sparsity, further enhancing the selection process by avoiding dependency of the selected model on the user-chosen model size.

In Chapter 4, we describe Batch Adapted Neighbour Searching (BANS), a real-time streaming feature selection method that employs dynamic batch processing and a one-step hard iteration threshold method. Through simulation and the application of BANS to real data, we showed that BANS is effective at retaining features with significant joint effects and we demonstrated its ability to handle complex dependencies among features while maintaining high performance at lower computation costs than other methods.

Overall, the statistical methods and software developed in this dissertation represent innovative steps toward solving some common challenges associated with Big Data through removing redundant features. We expect the **SMLE** R package, and the ISSE and BANS algorithms to be useful tools for researchers and practitioners across various scientific domains.

5.1 Future work

This thesis describes a software package and two novel algorithms, and demonstrates the superiority of these approaches over other approaches. This thesis lays the groundwork for several promising directions in high-dimensional statistical analysis and software tool enhancement. Future work will expand the utility of these approaches and refine the tools to better address the complexities of big data.

For **SMLE**, future development could focus on extending its application to other modeling scenarios beyond generalized linear models (GLMs). Potential areas include Cox's proportional hazards models and multivariate response regression, where SMLE-based methods have shown promise in the literature [45, 74, 75]. Additionally, integrating **SMLE** into a distributed computing framework would enable the package to handle big data scenarios more efficiently. This enhancement would utilize parallel computing resources to improve scalability and processing speed and ensure suitability for data stored across multiple locations or in streaming data environments.

The novel feature selection algorithm ISSE can be further developed to address a wider array of data analysis challenges. Future work includes refining the adaptive variant, aISSE, to enhance its ability to dynamically determine model sparsity. Integrating ISSE and aISSE into distributed computing environments will improve their scalability and efficiency, making them more suitable for large-scale high-dimensional datasets. Additional research can explore the application of ISSE in different domains, such as genomics and finance.

Finally, although BANS currently relies on a predefined parameter k for the number of

retained features, future work includes developing a strategy for the method to adaptively determine the best choice of k, further streamlining and automating the feature selection process. As with aISSE, we also plan to integrate BANS into a distributed computing environment to increase the scale of datasets it can handle and improve computational times.

By continuing to expand and refine these tools, future research can provide even more powerful and versatile solutions for ultrahigh-dimensional data, thereby enabling researchers and practitioners across various scientific domains to address increasingly complex problems.

References

[1] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

[2] Daniel Balikov, Adam Jacobson, and Lev Prasov. Glaucoma syndromes: Insights into glaucoma genetics and pathogenesis from monogenic syndromic disorders. *Genes*, 12(9), 2021.

[3] Emre Barut, Jianqing Fan, and Anne Verhasselt. Conditional sure independence screening. *Journal of the American Statistical Association*, 111(514):1051–1062, 2016.

[4] Thomas Blumensath and Mike E. Davies. Iterative thresholding for sparse approximations. *Journal of Fourier Analysis and Applications*, 14(5-6):629–654, December 2008.

[5] Thomas Blumensath and Mike E. Davies. Iterative hard thresholding for compressed sensing. *Applied and Computational Harmonic Analysis*, 27(3):4028–4031, 2009.

[6] Peter Bühlmann and Sara Van De Geer. *Statistics for high-dimensional data: Methods, theory and applications*. Springer Science & Business Media, 2011.

[7] Jiahua Chen and Zehua Chen. Extended Bayesian Information Criteria for model selection with large model spaces. *Biometrika*, 95(3):759–771, 09 2008.

[8] Apostolos Dimitromanolakis, Jingxiong Xu, Agnieszka Krol, and Laurent Briollais. **sim1000G**: A user-friendly genetic variant simulator in R for unrelated individuals and family-based designs. *BMC Bioinformatics*, 20(1):26, 2019.

[9] David L. Donoho. High-dimensional data analysis: The curses and blessings of dimensionality. In *AMS Math Challenges Lecture*, pages 1–32, 2000.

[10] D.L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

[11] Isabella Eigner and Andreas Hamper. *Predictive Analytics in Health Care: Methods and Approaches to Identify the Risk of Readmission*, pages 55–73. Springer International Publishing, Cham, 2018.

[12] Jianqing Fan, Yang Feng, and Runze Song. Nonparametric independence screening in sparse ultra-high-dimensional additive models. *Journal of the American Statistical Association*, 106(494):544–557, 2011.

[13] Jianqing Fan and Jinchi Lv. Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(5):849–911, 2008.

[14] Jianqing Fan and Jinchi Lv. Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society B*, 70(5):849–911, 2008.

[15] Jianqing Fan and Jinchi Lv. Ultra-high dimensional feature selection: beyond the linear model. *Journal of Machine Learning Research*, 10:2013–2038, 2009.

[16] Jianqing Fan and Jinchi Lv. A selective overview of variable selection in high-dimensional feature space. *Statistica Sinica*, 20(1):101–148, 2010.

[17] Jianqing Fan and Rui Song. Sure independence screening in generalized linear models with np-dimensionality. *The Annals of Statistics*, 38(6):3567–3604, 2010.

[18] Matteo Fasiolo. ***mvnfast****: An Introduction to mvnfast.*, 2016. R package version 0.1.6.

[19] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010.

[20] Harriett Fuller, Yiwen Zhu, Jayna Nicholas, Haley A. Chatelaine, Emily M. Drzymalla, Afrand K. Sarvestani, Sachelly Julin-Serrano, Usman A. Tahir, Nasa Sinnott-Armstrong, Laura M. Raffield, Ali Rahnavard, Xinwei Hua, Katherine H. Shutta, and Burcu F. Darst. Metabolomic epidemiology offers insights into disease aetiology. *Nature Metabolism*, 5(10):1656–1672, October 2023.

[21] Dimitrios Gherghel, Stephen Orgl, and Beatrice Dubler. Relationship between ocular perfusion pressure and retrobulbar blood flow in patients with glaucoma with progressive damage. *American Journal of Ophthalmology*, 130(5):597–605, 2000.

[22] Anders Gorst-Rasmussen and Thomas Scheike. Independent screening for single-index hazard rate models with ultrahigh dimensional features. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 75(2):217–245, 2013.

[23] Ibrahim Abaker Targio Hashem, Ibrar Yaqoob, Nor Badrul Anuar, Salimah Mokhtar, Abdullah Gani, and Samee Ullah Khan. The rise of "big data" on cloud computing: Review and open research issues. *Information Systems*, 47:98–115, 2015.

[24] Steven C. H. Hoi, Jialei Wang, Peilin Zhao, and Rong Jin. Online feature selection for mining big data. In *Proceedings of the 1st International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications*, BigMine '12, pages 93–100, New York, NY, USA, 2012. Association for Computing Machinery.

[25] Jialiang Huang, Ning Zhang, and Yixin Fang. Feature screening for ultrahigh-dimensional categorical data. *Journal of the American Statistical Association*, 109(507):1268–1279, 2014.

[26] David J. Hunter. Gene-environment interactions in human diseases. *Nature Reviews Genetics*, 6(4):287–298, 2005.

[27] Michael I. Jordan, Jason D. Lee, and Yun Yang. Communication-efficient distributed statistical inference. *Journal of the American Statistical Association*, 114(526):668681, 2019.

[28] Gang Kou, Yong Xu, Yi Peng, Feng Shen, Yang Chen, Kun Chang, and Shaomin Kou. Bankruptcy prediction for smes using transactional data and two-stage multiobjective feature selection. *Decision Support Systems*, 140:113429, 2021.

[29] Runze Li, Liying Huang, and John Dziak. *VariableScreening: High-Dimensional Screening for Semiparametric Longitudinal Regression*, 2022. R package version 0.2.1.

[30] Runze Li, Wei Zhong, and Liping Zhu. Feature screening via distance correlation learning. *Journal of the American Statistical Association*, 107(499):1129–1139, 2012.

[31] Runze Li, Wei Zhong, and Lixing Zhu. Feature screening via distance correlation learning. *Journal of the American Statistical Association*, 107(499):1129–1139, 2012.

[32] Xingxiang Li, Runze Li, Zhiming Xia, and Chen Xu. Distributed feature screening via componentwise debiasing. *Journal of Machine Learning Research*, 21(24):1–32, 2020.

[33] Jinghua Liu, Yaojin Lin, Shunxiang Wu, and Chenxi Wang. Online multi-label group feature selection. *Knowledge-Based Systems*, 143:42–57, 2018.

[34] JingYuan Liu, Wei Zhong, and RunZe Li. A selective overview of feature screening for ultrahigh-dimensional data. *Science China Mathematics*, 58(10):1–22, 2015.

[35] Ruqian Lu, Xiaolong Jin, Songmao Zhang, Meikang Qiu, and Xindong Wu. A study on big knowledge and its engineering issues. *IEEE Transactions on Knowledge and Data Engineering*, 31(9):1630–1644, 2018.

[36] Jason Macanian and Sansar C. Sharma. Pathogenesis of glaucoma. *Encyclopedia*, 2(4):1803–1810, 2022.

[37] James Manyika, Michael Chui, Brad Brown, Jacques Bughin, Richard Dobbs, Charles Roxburgh, and Angela Hung Byers. Big data: The next frontier for innovation, competition, and productivity, 2011. McKinsey Global Institute.

[38] Giulia Michelotti, Kelvin Wong, Vanessa Forgetta, Yihan Chen, Brent Richards, Josip Ivica, Divya Joshi, and Carolyn Balion. Metabolomic profiling on 9,992 participants

using ultra-performance liquid chromatography and mass spectrometer data support document. Technical report, Canadian Longitudinal Study on Aging, 2023.

[39] John Nelder and Robert Wedderburn. Generalized linear models. *Journal of the Royal Statistical Society: Series A (General)*, 135(3):370–384, 1972.

[40] Frederick Novomestky. ***matrixcalc****: Collection of Functions for Matrix Calculations*, 2022. R package version 1.0-6.

[41] AB Pant. *Vanillylmandelic Acid (VMA)*, pages 1036–1036. Springer Nature Singapore, Singapore, 2024.

[42] Andy Patrizio. Idc: Expect 175 zettabytes of data worldwide by 2025. *Network World*, 3, 2018.

[43] Simon Perkins and James Theiler. Online feature selection using grafting. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 592–599, 2003.

[44] Mary Playdon, Amit Joshi, Fred Kwadwo Tabung, Susan Cheng, Marcel Henglin, Anna Kim, Tiange Lin, Erik H. van Roekel, Jie Huang, Jan Krumsiek, Yudong Wang, Elizabeth A. Math, Marinella Temprosa, Scarlett C. Moore, Bo L. K. Chawes, Heather H. Eliassen, Andrea Gsur, Marc J. Gunter, Shuko Harada, Claudia Langenberg, Matej Oresic, Wei Perng, Woon-Puay Seow, and Ondine A. Zeleznik. Metabolomics analytics workflow for epidemiological research: Perspectives from the consortium of metabolomics studies (comets). *Metabolites*, 9(7):145, 2019.

[45] Lianqian Qu, Meilin Hao, and Liuquan Sun. Sparse composite quantile regression with ultra-high dimensional heterogeneous data. *Statistica Sinica*, 32:459–475, 2022.

[46] Wullianallur Raghupathi and Viju Raghupathi. Big data analytics in healthcare: promise and potential. *Health Information Science and Systems*, 2(1):3, 2014.

[47] Parminder Raina, Christina Wolfson, Susan Kirkland, Lauren E Griffith, Cynthia Balion, Benot Cossette, Isabelle Dionne, Scott Hofer, David Hogan, E R van den Heuvel, Teresa Liu-Ambrose, Verena Menec, Gerald Mugford, Christopher Patterson, Hlne Payette, Brent Richards, Harry Shannon, Debra Sheets, Vanessa Taler, Mary Thompson, Holly Tuokko, Andrew Wister, Changbao Wu, and Lynne Young. Cohort Profile: The Canadian Longitudinal Study on Aging (CLSA). *International Journal of Epidemiology*, 48(6):1752–1753j, 2019.

[48] Sergio Ramírez-Gallego, Héctor Mouriño-Talín, David Martínez-Rego, Verónica Bolón-Canedo, José Manuel Benítez, Amparo Alonso-Betanzos, and Francisco Herrera. An information theory-based feature selection framework for big data under apache spark. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 48(9):1441–1453, 2017.

[49] Maitreyi Rathod, Sushmita Chatterjee, Shruti Dutta, Rajiv Kalraiya, Dibyendu Bhattacharyya, and Abhijit De. Mannose glycosylation is an integral step for NIS localization and function in human breast cancer cells. *Journal of Cell Science*, 132(20):jcs232058, 10 2019.

[50] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2022.

[51] Christina Reese, Myungkyu Lee, and Xihong Lin. Feature screening for ultrahigh-dimensional categorical data. *Journal of the American Statistical Association*, 113(524):1765–1776, 2018.

[52] Diego Franco Saldana and Yang Feng. **SIS**: An R package for sure independence screening in ultrahigh-dimensional statistical models. *Journal of Statistical Software*, 83(2):1–25, 2018.

[53] Diego Franco Saldana and Yang Feng. SIS: An R package for sure independence screening in ultrahigh-dimensional statistical models. *Journal of Statistical Software*, 83(2):1–25, 2018.

[54] Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

[55] Avrum Spira, Jennifer E Beane, Vishal Shah, Katrina Steiling, Gang Liu, Frank Schembri, Sean Gilman, Yves-Martine Dumas, Paul Calner, Paola Sebastiani, Sriram Sridhar, John Beamis, Carla Lamb, Timothy Anderson, Norman Gerry, Joseph Keane, Marc E Lenburg, and Jerome S Brody. Airway epithelial gene expression in the diagnostic evaluation of smokers with suspect lung cancer. *Nature medicine*, 13(3):361366, March 2007.

[56] Xudong Sun, Yulin He, Dingming Wu, and Joshua Huang. Survey of distributed computing frameworks for supporting big data analysis. *Big Data Mining and Analytics*, 6(2):154–169, 2023.

[57] The 1000 Genomes Project Consortium. A global reference for human genetic variation. *Nature*, 526(7571):68–74, 2015.

[58] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society B*, 58(1):267–288, 1996.

[59] Stephen Turner. **qqman**: An R package for visualizing gwas results using q-q and manhattan plots. *The Journal of Open Source Software*, 2018.

[60] Peter Visscher, Naomi Wray, Qian Zhang, Pamela Sklar, Mark McCarthy, Matthew Brown, and Jian Yang. 10 years of gwas discovery: Biology, function, and translation. *American Journal of Human Genetics*, 101(1):5–22, Jul 2017.

[61] Hansheng Wang. Forward regression for ultra-high dimensional variable screening. *Journal of the American Statistical Association*, 104(488):1512–1524, 2009.

[62] Hansheng Wang. Forward regression for ultrahigh-dimensional variable selection. *Journal of the American Statistical Association*, 104(488):1512–1524, 2009.

[63] Hansheng Wang, Bo Li, and Chenlei Leng. Shrinkage tuning parameter selection in high-dimensional models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(3):671–683, 2009.

[64] Jianqing Wang and Chen Leng. High-dimensional ordinary least squares projection for screening variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(4):717–733, 2014.

[65] Jing Wang, Meng Wang, Peipei Li, Luoqi Liu, Zhongqiu Zhao, Xuegang Hu, and Xindong Wu. Online feature selection with group structure analysis. *IEEE Transactions on Knowledge and Data Engineering*, 27(11):3029–3041, 2015.

[66] Song Wang and Chen Leng. High-dimensional ordinary least-squares projection for screening variables. *Journal of the American Statistical Association*, 111(514):994–1008, 2016.

[67] Larry Wasserman and Kathryn Roeder. High-dimensional variable selection. *Annals of Statistics*, 37(5A):2178–2201, 2009.

[68] David S. Wishart, An Chi Guo, Eilis Oler, Francois Wang, Awais Anjum, Helen Peters, Robert Dizon, Zannatul Sayeeda, Song Tian, Brian Lee, Maksym Berjanskii, Richard Mah, Minoru Yamamoto, John Jovel, Carlos Torres-Calzada, Morgan Hiebert-Giesbrecht, Vivian W. Lui, Dorsa Varshavi, Daniel Allen, David Arndt, Navdeep Khetarpal, Arushan Sivakumaran, Kevin Harford, Sarah Sanford, Kelvin Yee, Xue Cao, Zane Budinski, Janika Liigand, Lianwen Zhang, Jing Zheng, Rajib Mandal, Nadezhda Karu, Maija Dambrova, Helgi B. Schith, Roman Greiner, and Vaibhav Gautam. HMDB 5.0: the Human Metabolome Database for 2022. *Nucleic Acids Research*, 50(D1):D622–D631, 2022.

[69] Xindong Wu, Kui Yu, Hao Wang, and Wei Ding. Online streaming feature selection. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 1159–1166, 2010.

[70] Yuanshan Wu and Guosheng Yin. Conditional quantile screening in ultrahigh-dimensional heterogeneous data. *Biometrika*, 102(1):65–76, 2015.

[71] Yuanshan Wu and Guosheng Yin. Conditional Quantile Screening in Ultrahigh-Dimensional Heterogeneous Data. *Biometrika*, 102(1):65–76, 02 2015.

[72] Wei Xie, Feida Zhu, Jing Jiang, Ee-Peng Lim, and Ke Wang. Topicsketch: Real-time bursty topic detection from twitter. *IEEE Transactions on Knowledge and Data Engineering*, 28(8):2216 2229, 2016.

[73] Chen Xu and Jiahua Chen. The sparse mle for ultrahigh-dimensional feature screening. *Journal of the American Statistical Association*, 109(507):1257–1269, 2014.

[74] Guangren Yang, Sumin Hou, Luheng Wang, and Yanqing Sun. Feature screening in ultrahigh-dimensional additive cox model. *Journal of Statistical Computation and Simulation*, 88(6):1117–1133, 2018.

[75] Guangren Yang, Ye Yu, Runze Li, and Ann Buu. Feature screening in ultrahigh dimensional cox's model. *Statistica Sinica*, 26(3):881–901, 07 2016.

[76] Kui Yu, Wei Ding, and Xindong Wu. Lofs: Library of online streaming feature selection. *Knowledge-Based Systems*, 113, 03 2016.

[77] Kui Yu, Xindong Wu, Wei Ding, and Jian Pei. Scalable and accurate online feature selection for big data. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 11(2):1–39, 2016.

[78] Kui Yu, Xindong Wu, Wei Ding, and Jian Pei. Scalable and accurate online feature selection for big data. *ACM Trans. Knowl. Discov. Data*, 11(2), dec 2016.

[79] Qianxiang Zang, Chen Xu, and Kelly Burkett. Smle: An r package for joint feature screening in ultrahigh-dimensional glms, 2022.

[80] Qingchen Zhang, Laurence T Yang, Zhikui Chen, and Peng Li. A survey on deep learning for big data. *Information Fusion*, 42:146–157, 2018.

[81] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. Deep learning based recommender system: A survey and new perspectives. *ACM computing surveys (CSUR)*, 52(1):1–38, 2019.

[82] Jing Zhou, Dean Foster, Robert Stine, and Lyle Ungar. Streaming feature selection using alpha-investing. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 384–393, 2005.

[83] Jing Zhou, Dean P Foster, Robert A Stine, Lyle H Ungar, and Isabelle Guyon. Streamwise feature selection. *Journal of Machine Learning Research*, 7(9), 2006.

[84] Peng Zhou, Peipei Li, Shu Zhao, and Xindong Wu. Feature interaction for streaming feature selection. *IEEE Transactions on Neural Networks and Learning Systems*, 32(10):4691–4702, 2021.

[85] Tingyou Zhou, Liping Zhu, Chen Xu, and Runze Li. Model-free forward screening via cumulative divergence. *Journal of the American Statistical Association*, 115(531):1393–1405, 2020.

[86] Jin Zhu, Xueqin Wang, Liyuan Hu, Junhao Huang, Kangkang Jiang, Yanhang Zhang, Shiyun Lin, and Junxian Zhu. **abess**: A fast best-subset selection library in Python and R. *Journal of Machine Learning Research*, 23(202):1–7, 2022.

[87] Junxian Zhu, Canhong Wen, Jin Zhu, Heping Zhang, and Xueqin Wang. A polynomial algorithm for best-subset selection problem. *Proceedings of the National Academy of Sciences*, 117(52):33117–33123, 2020.

[88] Liping Zhu, Lexin Li, Runze Li, and Lixing Zhu. Model-free feature screening for ultrahigh-dimensional data. *Journal of the American Statistical Association*, 106(496):1464–1475, 2011.

[89] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.