# IMPROVED VARIABLE SELECTION WITH SECOND-GENERATION P-VALUES

By

Yi Zuo

Dissertation

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

in

Biostatistics

January 31, 2022

Nashville, Tennessee

Approved:

Qingxia (Cindy) Chen, Ph.D.

Jeffrey D. Blume, Ph.D.

Matthew S. Shotwell, Ph.D.

Thomas G. Stewart, Ph.D.

Yuankai Huo, Ph.D.

*In the vastness of probabilistic space and immensity of time, it is my joy to spend an instance and an epoch with you, Manyuan Zhang.*

## ACKNOWLEDGMENTS

I feel extremely fortunate to have the opportunity to pursue my doctoral degree at Vanderbilt University.

I would like to thank Dr. Jeffrey D. Blume, my advisor, for all his guidance, kindness, and humility. He inspired me to become not only a brilliant researcher, but a better person. I would like to thank Drs. Qingxia (Cindy) Chen, Matthew S. Shotwell, Thomas G. Stewart, and Yuankai Huo, for serving on my committee and providing invaluable advice to improve the work. I would like to thank Drs. Christopher Fonnesbeck, Zachary E. Warren, and Brett M. Kroncke for memorable collaborative experience. I would like to thank all faculty members that I met at Biostatistics Department, and Vanderbilt University at large, for their expertise and professionalism.

I would like to thank my cohort, Yue Gao, Chiara Di Gravio, Rebecca Irlmeier, Ryan Moore, Julia Thome, and Yan Yan. It is always fun to study and hang out with you guys. I would like to thank all the friends that I made in the past four years in Nashville.

I would like to thank my girlfriend, Manyuan Zhang, who has been nothing but supportive. You have always been my best friend, great companion, loved, encouraged, and entertained. I can not wait to come back and start a new chapter in life with you.

Last but not least, I want to thank my parents, Hong Chen and Hongbin Zuo, for their unconditional love, timely encouragement, and endless patience. You helped me reach this milestone, for whom I will always be grateful.

## TABLE OF CONTENTS

## LIST OF TABLES

## LIST OF FIGURES

# CHAPTER 1

## Variable Selection in Linear Models with Second-Generation P-Values

**Abstract**

Many statistical methods have been proposed for variable selection in the past century, but few balance inference and prediction tasks well. Here we report on a novel variable selection approach called Penalized regression with Second-Generation P-Values (ProSGPV). It captures the true model at the best rate achieved by current standards, is easy to implement in practice, and often yields the smallest parameter estimation error. The idea is to use an $\ell_0$ penalization scheme with second-generation p-values (SGPV), instead of traditional ones, to determine which variables remain in a model. The approach yields tangible advantages for balancing support recovery, parameter estimation, and prediction tasks. The ProSGPV algorithm can maintain its good performance even when there is strong collinearity among features or when a high dimensional feature space with $p > n$ is considered. We present extensive simulations and a real-world application comparing the ProSGPV approach with smoothly clipped absolute deviation (SCAD), adaptive lasso (AL), and minimax concave penalty with penalized linear unbiased selection (MC+). While the last three algorithms are among the current standards for variable selection, ProSGPV has superior inference performance and comparable prediction performance in certain scenarios. Supplementary materials are available online.

## 1.1 Introduction

Data are typically comprised of an outcome and features (predictors or covariates). A common scientific task is to separate important features (signals) from unrelated features (statistical noise) to facilitate modeling, learning, clinical diagnosis, and decision-making. Statistical models are selected for a variety of reasons: predictive ability, interpretability, ability to perform parameter inference, and ease of computation. A model's set of features is called its "support" and the task of recovering the model's true support from observed data is called "support recovery". A desirable variable selection method will tend to return the set of true predictors - i.e. those features with truly non-zero coefficients - with high probability. Support recovery aids inference, because knowing the model's true support benefits parameter estimation by reducing bias and improving efficiency. While an incorrectly specified model can sometimes have better predictive performance than a correctly specified model (Shmueli et al., 2010), having the correct support is essential for achieving optimal statistical inference (Zhang et al., 2009; Shortreed and Ertefaie, 2017).

Penalized likelihood procedures, originally optimized for prediction tasks, are widely used for variable selection. The lasso, an $\ell_1$ penalization method, produces models with strong predictive ability (Tibshirani, 1996). However, the lasso solution that maximizes predictive ability does not always lead to consistent support recovery (Leng et al., 2006; Meinshausen et al., 2006; Shmueli et al., 2010; Bogdan et al., 2015). This is because noise variables are often included in the lasso solution that maximizes predictive ability (Meinshausen et al., 2006). The adaptive lasso (AL), which introduces weights in the $\ell_1$ penalty, was proposed to resolve the issue that lasso solutions can be variable selection inconsistent (Zou, 2006). With clever choice of tuning parameters, and in large samples, the adaptive lasso can recover the true support with high probability and yield parameter estimates that converge properly (Zou, 2006). Smoothly clipped absolute deviation (SCAD) (Fan and Li, 2001) and minimax concave penalty with penalized linear unbiased selection (MC+) (Zhang et al., 2010) make use of distinctive piecewise linear thresholding functions to bridge the gap between the $\ell_0$ and $\ell_1$ algorithms. Both SCAD and MC+ seek to preserve large coefficients, like the $\ell_0$ penalty does, and shrink small coefficients, like the $\ell_1$ penalty does. While their variable selection properties have been well established, these methods are still not widely used in routine practice.

All of the above approaches place a strong emphasis on predictive ability, at the cost of subsequent inference tasks. Because inference is an essential component of scientific investigations, a variable selection approach that balances prediction and inference tasks is highly desirable. Since traditional p-values do not reflect whether a variable is scientifically relevant or not (Heinze et al., 2018), we investigated whether using second-generation p-values (SGPV) (Blume et al., 2018, 2019) would lead to good support recovery and subsequent parameter estimation and prediction. SGPVs emphasize scientific relevance in addition to statistical

significance, and thus they are a good tool for screening out noise features and identifying the true signals in a set of candidate variables.

Following this idea, we propose a variable selection algorithm based on an $\ell_0$-Penalized regression with SGPVs (ProSGPV). The ProSGPV algorithm has a high support recovery rate and low parameter estimation bias, while maintaining good prediction performance even in the high-dimensional setting where $p > n$. In a series of comprehensive simulations and a real-world application, the ProSGPV algorithm is shown to be a viable alternative to, and often a noticeable improvement on, current variable selection standards such as AL, SCAD and MC+. While only linear models are discussed in this paper, forthcoming work will show that the ProSGPV approach generalizes to models of other classes, including logistic regression, Poisson regression, Cox proportional hazards model, etc.

The structure of this paper is as follows. Section 1.2 provides a brief background. Section 1.3 describes the proposed ProSGPV algorithm. Section 1.4 presents simulation studies comparing ProSGPV to AL, SCAD, and MC+ under various feature correlation structures and signal-to-noise ratios. Section 1.5 illustrates the ProSGPV algorithm using a real-world data application. Section 1.6 discusses the practical implications of the simulation results and some limitations of ProSGPV, and summarizes key findings in the paper.

### 1.2 Background material

We review some fundamental ideas related to shrinkage, thresholding, inference, and prediction in the variable selection context to facilitate subsequent discussions about ProSGPV. Readers familiar with standard variable selection notation, lasso (section 1.2.1), adaptive lasso (section 1.2.2), SCAD and MC+ (section 1.2.3), and second-generation p-values (section 1.2.4) may skip to section 1.3 for the development of the ProSGPV algorithm.

#### 1.2.1 Lasso

The lasso is an $\ell_1$ penalization procedure and one of the most widely used regularization methods for prediction modeling (Tibshirani, 1996). It reduces the feature space and identifies a subset of features that maximize predictive accuracy subject to a sparsity condition induced by the $\ell_1$ penalty. The set of features selected by lasso is called the active set.

Let $Y = (Y_1, Y_2, ...Y_n)$ denote the response vector, $X$ denote the $n \times p$ design matrix, and $\beta \in \mathbb{R}^p$ denote the coefficient vector. $\lambda > 0$ is a regularization parameter. $||\cdot||_2^2$ is the squared $\ell_2$-norm and $||\cdot||_1$ is the $\ell_1$-norm.

Formally, the lasso solution is written as

$$\hat{\beta} = \arg\min_{\beta}\{\frac{1}{2}||Y - X\beta||_2^2 + \lambda||\beta||_1\} \tag{1.1}$$

The lasso is often used for variable selection because its solution encourages sparsity in the active set. However, even in the classical setting of a fixed $p$ and a growing $n$, the lasso active set tends to be different from the set of true signals. An exception to this is when true feature columns are roughly orthogonal to noise feature columns (Knight and Fu, 2000), which unfortunately, is seldom seen in practice. Wainwright, 2009b improved the ability of the lasso solution to recover the true support under random Gaussian designs and showed that lasso can recover the true support when the effect size is sufficiently large and when no noise variables are highly correlated with true features. However, these conditions are strong and hard to apply in practice. In addition, even when they are met, there is no explicit way to implement the procedure because the shrinkage factor $\lambda$ that yields the correct support recovery is unknown (Wang et al., 2013). Lastly, the soft thresholding function in lasso shrinks large effects and results in biased parameter estimates that are ideal for prediction tasks, but not necessarily optimal for inference tasks.

### 1.2.2 Adaptive lasso

The adaptive lasso (AL) uses weights in the $\ell_1$ penalty to address the inconsistent variable selection property of the lasso (Zou, 2006). With the right shrinkage parameter, initial weights, and weight moments, the adaptive lasso can recover the true support with high probability while preserving prediction performance. Formally, the solution to the adaptive lasso is:

$$\hat{\beta}^n = \arg\min_{\beta}\{\frac{1}{2}||Y - X\beta||_2^2 + \lambda_n||\hat{\omega}\beta||_1\} \tag{1.2}$$

where $\hat{\omega} = 1/|\hat{\beta}^*|^\gamma$. Here $\gamma > 0$ is a tuning parameter and $\hat{\beta}^*$ is any root-$n$-consistent estimator of the parameter $\beta$, for example, an OLS estimator, or a lasso estimator.

Zou, 2006 showed that AL has large-sample oracle (optimal) properties for support recovery and parameter estimation as $\lambda_n/\sqrt{n} \to 0$ and $\lambda_n n^{(\gamma-1)/2} \to \infty$. However, when the sample size is finite, it can be hard to find a combination of $\hat{\beta}^*$, $\gamma$, and $\lambda_n$ such that the resulting active set matches the true support and the estimated coefficients have low bias.

### 1.2.3 SCAD and MC+

SCAD and MC+ were designed to bridge $\ell_0$ and $\ell_1$ penalization schemes. As a result, both algorithms use nonconvex penalties. There are considerable advantages that come with using nonconvex penalization,

such as a sparse solution and reduced parameter estimation bias, see Fan and Lv, 2011, 2013; Zheng et al., 2014; Loh and Wainwright, 2015.

The penalty function in the SCAD corresponds to a quadratic spline function with knots at $\lambda$ and $\gamma\lambda$ (Fan and Li, 2001). With proper choice of regularization parameters, SCAD can yield consistent variable selection in large samples (Fan and Li, 2001). MC+ has two components: a minimax concave penalty (MCP) and a penalized linear unbiased selection (PLUS) algorithm (Zhang et al., 2010). MC+ returns a continuous piecewise linear path for each coefficient as the penalty increases from zero (least squares) to infinity (null model). When the penalty level is set to $\lambda = \sigma\sqrt{(2/n)\log(p)}$, the MC+ algorithm has a high probability of support recovery and does not need to assume the strong irrepresentable condition (Wainwright, 2009b) that is required by lasso for support recovery (Zhang et al., 2010). For visualization, Figure 1.1 displays the thresholding functions of $\ell_0$ and $\ell_1$ penalties, SCAD, and MC+ when the feature columns are orthogonal.

### 1.2.4 Second-generation p-values

Second-generation p-values (SGPV), denoted as $p_\delta$, were proposed for use in high dimensional multiple testing contexts (Blume et al., 2018, 2019). SGPVs attempt to resolve some of the deficiencies of traditional p-values by replacing the point null hypothesis with a pre-specified interval null $H_0 = [-\delta, \delta]$. The idea is to use the interval as a buffer region between "null" and "non-null" effects. The interval represents the set of effects that are scientifically indistinguishable or immeasurable from the point null due to limited precision or practicality. SGPV are essentially the fraction of data-supported hypotheses that are null, or nearly null, hypotheses.

Formally, let $\theta$ be a parameter of interest, and let $I = [\theta_l, \theta_u]$ be an interval estimate of $\theta$ whose length is given by $|I| = \theta_u - \theta_l$. In this paper we will use a 95% CI for $I$, but any type of the uncertainty interval can be used. If we denote the length of the interval null by $|H_0|$, then the SGPV $p_\delta$ is defined as

$$p_\delta = \frac{|I \cap H_0|}{|I|} \times \max\left\{\frac{|I|}{2|H_0|}, 1\right\} \tag{1.3}$$

where $I \cap H_0$ is the intersection of two intervals. The correction term $\max\{|I|/(2|H_0|), 1\}$ applies when the interval estimate is very wide, i.e., when $|I| > 2|H_0|$. In that case, the data are often inconclusive and the correction term shrinks the SGPV back to 1/2. As such, SGPVs indicate when data are compatible with null hypotheses ($p_\delta = 1$), or with alternative hypotheses ($p_\delta = 0$), or when data are inconclusive ($0 < p_\delta < 1$).

By design, SGPVs emphasize effects that are scientifically meaningful as defined by exceeding a pre-specified effect size $\delta$. Empirical studies have shown that SGPVs have the potential for identifying feature importance in high dimensional settings (Blume et al., 2018, 2019). This idea dovetails well with the natural

tendency in variable selection to keep variables whose effects are above some threshold, say $\delta$. One extension here is that we will let the null bound $\delta$ shrink to zero at a pre-specified rate. This slight modification of the basic SGPV idea makes variable selection by SPGVs much more effective. Sensitivity to the choice of the null bound is assessed in Section 1.3.4.

## 1.3 The ProSGPV algorithm

The ProSGPV algorithm is a two-stage algorithm. In the first stage, a candidate set of variables is acquired. In the second stage, an SGPV-based thresholding is applied to select variables from the candidate set that are meaningfully associated with the outcome.

### 1.3.1 Steps

The steps of the ProSGPV algorithm are shown below in Algorithm 1.

**Algorithm 1** ProSGPV

**procedure** PROSGPV($X$, $Y$)
  **Stage one**: Find a candidate set
    Standardize all inputs (the outcome and features)
    Fit a lasso and find $\lambda_{\text{gic}}$ using generalized information criterion
    Fit an OLS model on the lasso active set
  **Stage two**: SGPV screening
    Extract the confidence intervals of all variables from the previous OLS model
    Calculate the mean coefficient standard error $\overline{SE}$ of standardized features
    Get the SGPV for each variable $k$ with $I_k = \hat{\beta}_k \pm 1.96 \times SE_k$ and $H_0 = [-\overline{SE}, \overline{SE}]$
    Keep variables with SGPV of zero
    Re-run the OLS model with selected variables on the original scale
**end procedure**

Note that the outcome and features are standardized except for the final step. Generalized information criterion (GIC) (Fan and Tang, 2013) is used to find the shrinkage parameter $\lambda_{\text{gic}}$ that leads to a fully-relaxed lasso (Meinshausen, 2007) in the first stage. $\lambda$ could also be found through cross-validation, as there is evidence that a range of $\lambda$s will lead to the true support (Fan and Li, 2001; Zou, 2006; Wang et al., 2013; Sun et al., 2019). That adds to the flexibility of the algortihm. In the second stage, SGPVs are used to screen variables in the candidate set, where the null bound $\delta$ is derived from coefficient standard errors. Sensitivity to the choice of the null bound $\delta$ is assessed in Section 1.3.4. We have implemented the ProSGPV algorithm

in the **ProSGPV** R package, which is available from the Comprehensive R Archive Network (CRAN) at https://CRAN.R-project.org/package=ProSGPV.

### 1.3.2 Solution

The solution to the ProSGPV algorithm $\hat{\beta}^{\text{pro}}$ is

$$\begin{gathered}\hat{\beta}^{\text{pro}} = \hat{\beta}^{\text{ols}}_{|S} \in \mathbb{R}^p, \text{ where} \\ S = \{k \in C : |\hat{\beta}^{\text{ols}}_k| > \lambda_k\}, C = \{j \in \{1,2,...,p\} : |\hat{\beta}^{\text{lasso}}_j| > 0\}\end{gathered} \tag{1.4}$$

where $\hat{\beta}^{\text{ols}}_{|S}$ is a vector of length $p$ with non-zero elements being the OLS coefficient estimates from the model with variables only in the set $S$, the final selection set. $C$ is the candidate set from the first-stage screening. $\hat{\beta}^{\text{lasso}}_j$ is the $j$th lasso solution evaluated at $\lambda_{\text{gic}}$ in the first stage. In the second stage, the cutoff is $\lambda_k = 1.96 \times SE_k + \overline{SE}$ and $\lambda_k$ is constant over $k$ when the features are all centered and standardized. In that case, the coefficient standard errors are identical.

The ProSGPV algorithm is effectively a hard thresholding function. In the first stage, variables not selected by lasso are shrunk to zero. In the second stage, ProSGPV relaxes the coefficients and shrinks effects smaller than $\overline{SE}$ to zero while preserving large effects. Because this is a two-stage algorithm, there does not appear to be a simple closed-form solution for the implied thresholding without conditioning on the first stage. However, this would-be threshold, call it $\lambda_{\text{new}}$, tends to be larger than $\lambda_{\text{gic}}$ from lasso. The only routine exception to this is when data have weak signals or high correlation. But in that case, no algorithm can fully recover the true support (Zhao and Yu, 2006; Wainwright, 2009a) .

A visualization of thresholding functions for several penalization methods (assuming orthogonal features) is displayed in Figure 1.1. The hard thresholding function in the panel (1) shrinks the coefficient estimates to zero when the effects are less than $\lambda$ and preserves them otherwise. The lasso in (2) shrinks small effects to zero and shrinks large effects by $\lambda$. SCAD and MCP in (3) bridge the gap between a hard thresholding function seen in (1) and a soft thresholding function in (2). When the coefficient is small ($|\hat{\theta}| \leq \lambda$), both methods have the same behavior as the lasso because the coefficient is shrunk to zero in all cases. When the coefficient is large ($|\hat{\theta}| \geq \gamma\lambda$), SCAD and MCP have the same behavior as the hard thresholding (no shrinkage is applied). What distinguishes SCAD and MCP is the shape of its thresholding function between $\lambda$ and $\gamma\lambda$. As mentioned earlier, ProSGPV amounts to a hard thresholding function whose cutoff is usually larger than $\lambda$.

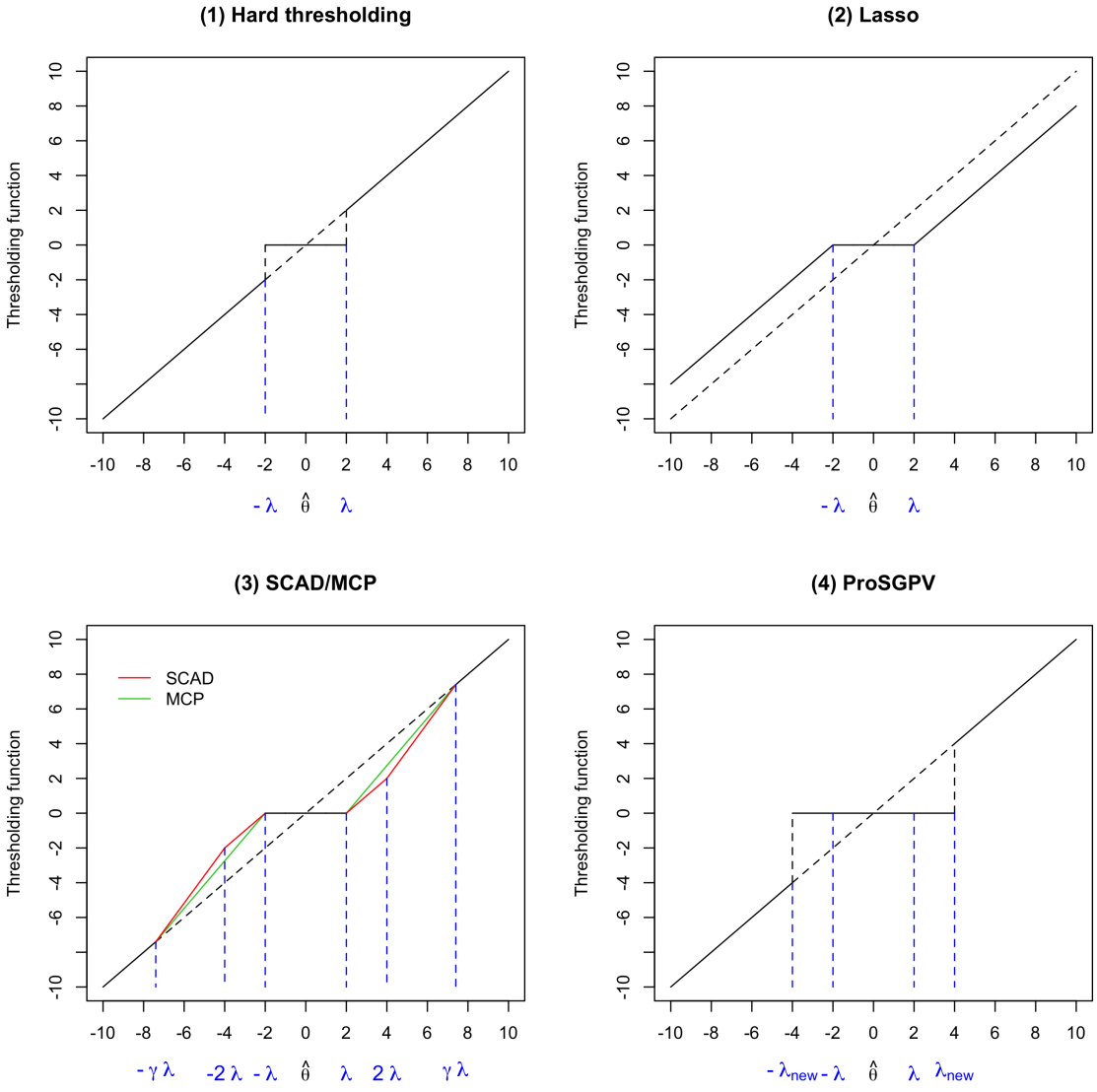


Figure 1.1: Thresholding functions from five algorithms when features are orthogonal.

### 1.3.3 Example

Figure 1.2 shows the effect of the ProSGPV algorithm on the regression coefficients in our simulated setting. Suppose that the true data-generating model is $y = X\beta + \varepsilon$ where $y$ is a vector of length 400. The design matrix $X$ has five columns with mean zero and covariance matrix $\Sigma_{i,j} = 0.5^{|i-j|}$. The coefficient vector $\beta$ is zero everywhere except $\beta_3 = 0.28$. The errors are i.i.d. $N(0,1)$. We see in Figure 1.2 that the ProSGPV algorithm succeeds by selecting V3 whereas the lasso and relaxed lasso select V3 and V5 at $\lambda_{\text{gic}}$.

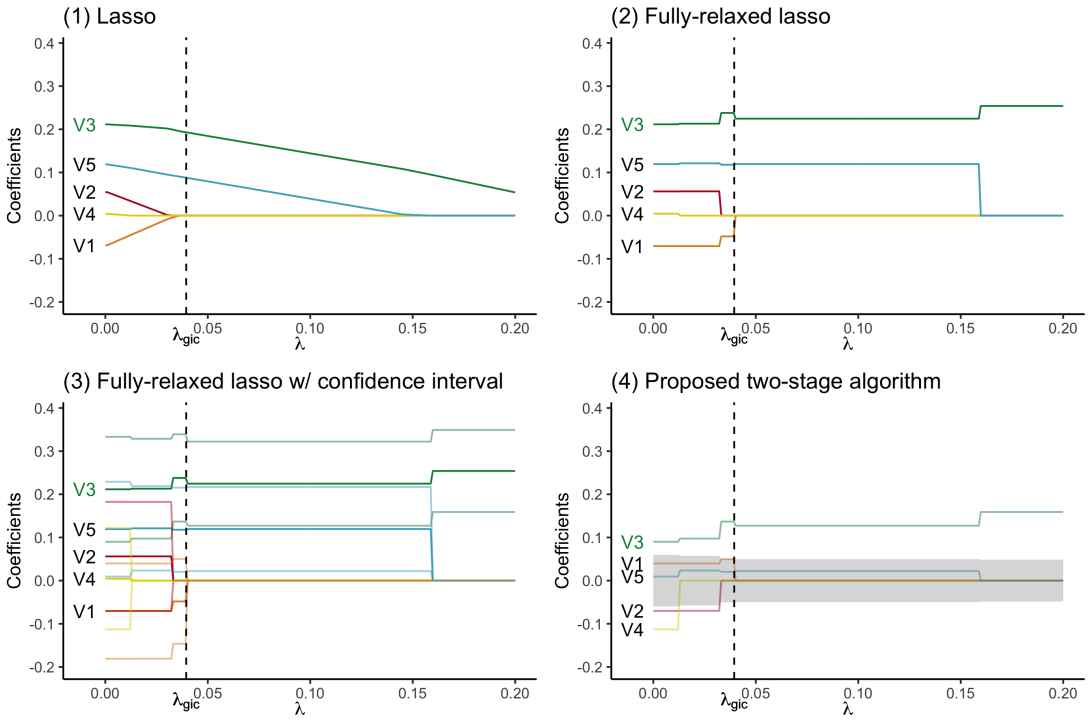


Figure 1.2: Illustration of the ProSGPV algorithm. Panel (1) presents the colored lasso solution path where the vertical dotted line is the $\lambda_{\text{gic}}$. Panel (2) shows the fully-relaxed lasso path with point estimates only. Panel (3) shows the same path plus 95% confidence intervals in light colors. Panel (4) is the proposed two-stage algorithm's selection path. The shaded area is the null region and only the 95% confidence bound that is closer to zero is shown for each variable.

### 1.3.4 Null bound

The null bound in ProSGPV is set to be the average coefficient standard error, say $\overline{SE}$, from the OLS model on the lasso candidate set. Because of the scaling, this is equivalent to hard-thresholding variables whose absolute coefficients are below $1.96 \times SE_k + \overline{SE} \approx 3 \times \overline{SE}$. This is in line with variable selection ideas from literature. Fan and Li, 2006 argued that in order to achieve optimal properties of variable selection, the amount of lasso shrinkage must be proportional to the standard error of the maximum likelihood estimates of coefficients. Intuitively, the interval null acts as a buffer zone to screen out effects that are likely false discoveries. By definition, the sampling distribution of false discoveries will be near the point null (since they are "false" discoveries) and the variance of this distribution shrinks at a rate proportional to the information in the sample. Hence, using the SE to delineate the smallest effect size of interest is natural. It is possible that a constant multiplier of the SE might yield a better Type I-Type II error tradeoff, but after trying some obvious variations we did not find anything better.

A sensitivity analysis on the choice of the null bound was conducted and is summarized in Figure 1.3.

We compared the support recovery performance of ProSGPV using different null bounds when signal-to-

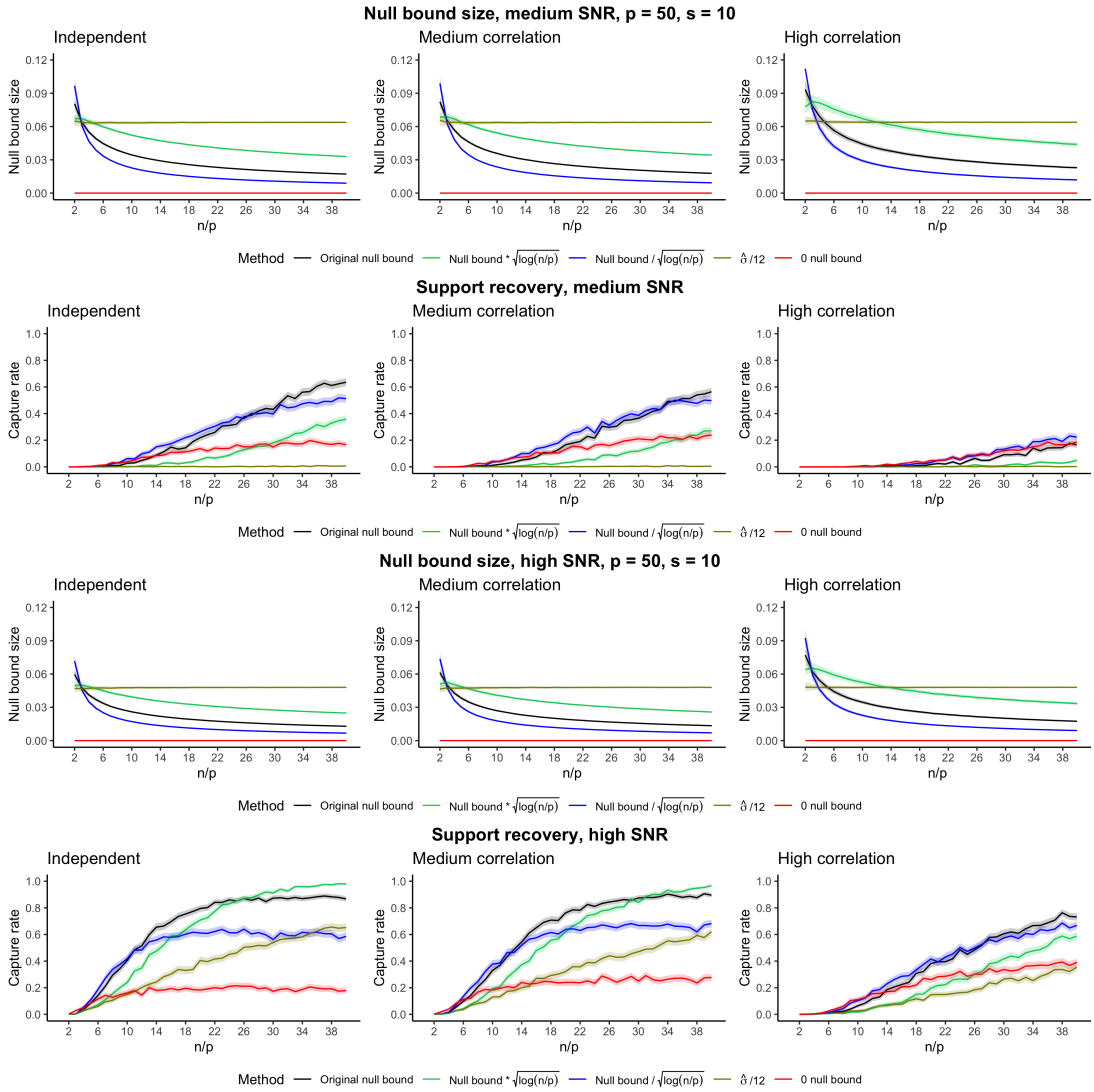


Figure 1.3: Sensitivity of the null bound in the ProSGPV algorithm. The size of the null bound and the capture rate of the exact true model from ProSGPV with different null bounds are compared under medium and high signal-to-noise ratios. Choices of the null bound include the original bound $\overline{SE}$, $\overline{SE} * \sqrt{\log(n/p)}$, $\overline{SE}/\sqrt{\log(n/p)}$, $\hat{\sigma}/12$, and 0. The top plot shows how the null bound (median, first and third quartiles) changes with $n$, and the bottom plot shows the capture rates surrounded by 95% Wald confidence intervals over 1000 simulations.

noise ratio (SNR) is medium or high and when $n > p$. Choices of null bounds include the original bound $\overline{SE}$, $\overline{SE} \times \sqrt{\log(n/p)}$, $\overline{SE}/\sqrt{\log(n/p)}$, $\hat{\sigma}/12$, and 0. When the null bound is constant, e.g., $\hat{\sigma}/12$, the support recovery performance is poor. When the null bound is scaled by $\sqrt{\log(n/p)}$, performance appears to be slightly improved in the high correlation case, but, importantly, is inferior in all other cases. When the null bound is set at 0, ProSGPV amounts to selecting variables using traditional p-values. In this case, the support recovery performance is expectedly poor even when SNR is high, because the null bound of 0 leads to many

false positives (Kaufman and Rosset, 2014; Janson et al., 2015). When $p > n$, the above observations hold because the null bound is calculated from a model with a reduced number of features (same order as $s << p$, where $s$ is the number of true signals). This sparsity assumption is necessary for successful high-dimensional support recovery (Meinshausen et al., 2006; Zhao and Yu, 2006; Wainwright, 2009a). Hence, allowing the null bound, which acts as a thresholding function, to shrink at a $\sqrt{n}$-rate, appears to offer the best performance across the widest range of scenarios.

### 1.3.5 Similar algorithms from the literature

Other two-stage algorithms have been proposed for pre-screening features (Meinshausen et al., 2009; Zhang et al., 2009; Wasserman and Roeder, 2009; Zhou, 2009, 2010; Sun et al., 2019; Weng et al., 2019; Wang et al., 2020). Meinshausen et al., 2009 proposed a two-stage thresholded lasso, where a lasso model is fit and features are kept if they pass a data-dependent coefficient threshold. Because of this, the resulting coefficient estimates are biased even when the correct support is recovered. Wasserman and Roeder, 2009 proposed using variable selection methods (lasso, marginal regression, forward stepwise regression, etc.) with cross-validation to pre-screen candidate variables before using Bonferroni corrected t-tests to identify and remove noise features. Wasserman's method controls the Type I error rate across all features, but pays a higher price in false negatives. ProSGPV, however, allows the Type I error rate to shrink towards zero and yields fewer false positives (See Supplementary Figure 3). Zhang et al., 2009 identified relevant and irrelevant features from lasso in the first stage and fit another $\ell_1$-penalized regression using only irrelevant features afterwards. However, Zhang et al., 2009 emphasizes parameter estimation and neglects support recovery. In addition, their algorithm needs to run multiple cross-validations while ProSGPV uses GIC to tune $\lambda$ and is therefore much faster to compute. Sun et al., 2019 proposed the hard thresholding regression (HRS). When lasso is used to derive initial weights, the HRS reduces to the fully relaxed lasso, which is the first stage of our two-stage ProSGPV algorithm. Unlike our algorithm, HRS keeps all variables that survive the first stage. Lastly, Zhou, 2009, 2010 used lasso or the Dantzig selector to pre-screen and then used a fully relaxed model on thresholded coefficients with a data-driven bound; Weng et al., 2019 selected important variables and penalized only the unselected variables for the final variable selection; Wang et al., 2020 used a bridge regression in the first stage and thresholded variables in the second stage.

### 1.3.6 Special case: one-stage ProSGPV algorithm

When $\lambda_{\text{gic}}$ is replaced with zero in the first stage of lasso, ProSGPV reduces to a one-stage algorithm. That amounts to calculating the SGPV for each variable in the full OLS model and selecting ones that are above the threshold. The one-stage ProSGPV is faster to compute, as no lasso solution path is required. However,

it does not appear to be variable selection consistent in the limit, and its inferential performance is inferior to that of the two-stage ProSGPV when data do not contain strong signals or features are highly correlated. Moreover, it is not applicable when $p > n$, i.e., when the OLS model is not identifiable. For completeness, the support recovery performance of the one-stage algorithm can be found in Figure 1.4. Its performance is very close to the two-stage algorithm when explanatory variables are independent.

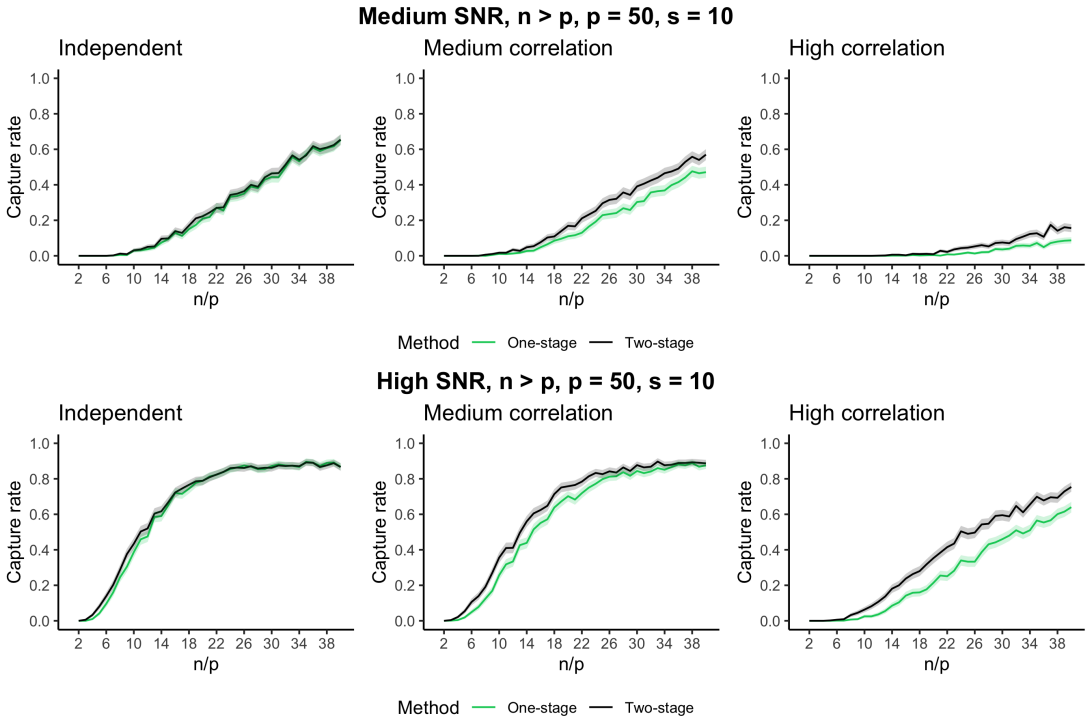


Figure 1.4: Support recovery rates of one-stage and two-stage ProSGPV algorithms. Each curve represents the capture rate of the exact true model over 1000 simulations. The two-stage algorithm selects the $\lambda$ that minimizes the generalized information criterion in the first stage. Shaded belts are 95% Wald confidence intervals.

### 1.3.7 Summary

The ideas behind the ProSGPV algorithm are intuitive: exclude small effects using a data-dependent threshold for noise and keep large effects. ProSGPV is essentially an $\ell_0$-penalized regression. Unlike the $\ell_1$ penalty, $\ell_0$ optimization is nonconvex, so it is harder to compute and less popular in practice. However, our algorithm avoids enumerating all possible combinations of variables by leveraging the lasso solution in the first stage and threshold effects with an explicit bound afterwards. That translates into less computational cost than other convex optimization algorithms (as seen in Figure 1.11). ProSGPV can also be thought of as a variation of the thresholded lasso with refitting. van de Geer et al., 2011 showed that the thresholded lasso with refitting requires less severe minimal signal conditions for successful support recovery than adaptive lasso.

While lasso is used in the first stage screening, other variable selection methods, such as Sure Independence Screening (SIS) (Fan and Lv, 2008), can be used there. This adds the /flexibility to our algorithm. Lastly, in terms of post-selection inference, the point estimates and corresponding confidence intervals derived from our algorithm are best when the selected model matches the true underlying model. Even when ProSGPV misses true signals, those missed variables often have small effects, which results in minimal impact on the inference of the other larger effects.

### 1.4 Simulation studies

Extensive simulation studies were conducted to evaluate the inferential and prediction performance of the ProSGPV algorithm and compare it to existing methods. We investigated both traditional $n > p$ and high-dimensional $p > n$ settings.

#### 1.4.1 Design

The simulation setup is motivated by similar investigations such as Hastie et al., 2020. We set sample size $n$, dimension of explanatory variables $p$, sparsity level $s$ (number of true signals), true coefficient vector $\beta_0 \in \mathbb{R}^p$, autocorrelation level $\rho$ within explanatory variables, and signal-to-noise ratio (SNR) $\nu$.

In the traditional $n > p$ setting, $p$ is fixed at 50 and $n$ ranges from 100 to 2000 with an increment of 50. The number of true signals $s$ is fixed at 10. In the high-dimensional setting, $n$ is fixed at 200 and $p$ ranges from 200 to 2000 with an increment of 20. Here, the number of true signals is fixed at 4. $\beta_0$ has $s$ non-zero values equally-spaced between one and five, at random positions, and the rest are zero. The coefficients are half positive and half negative. $\rho$ can take the value of 0 (independent), 0.35 (medium autocorrelation), and 0.7 (high autocorrelation). SNR is defined as SNR $= Var(f(x))/Var(\varepsilon)$, where data are generated from a probabilistic distribution. SNR take the value of 0.7 (moderate SNR), and 2 (high SNR) (Hastie et al., 2020). We evaluated the performance of each algorithm using standard metrics: support recovery rate, Type I error rate, power, false discovery rate, false non-discovery rate, along with the mean absolute error (defined below) for parameter estimation, prediction accuracy in a separate test set, and running time. See Table 1.1 for detailed definitions of the metrics for inference.

**Step 1**: Draw $n$ rows of the matrix $X \in \mathbb{R}^{n\times p}$ i.i.d. from $N_p(0,\Sigma)$, where $\Sigma \in \mathbb{R}^{p\times p}$ has entry $(i,j)$ equal to $\rho^{|i-j|}$.

**Step 2**: Generate the response vector $Y \in \mathbb{R}^n$ from $N_n(X\beta_0, \sigma^2 I)$, with $\sigma^2$ defined to meet the desired SNR level $\nu$, i.e., $\sigma^2 = \beta_0^T \Sigma \beta_0 / \nu$.

**Step 3**: Run SCAD, MC+, AL, and ProSGPV on the training set with $n$ observations; record the active set from each algorithm; compute evaluation metrics in Table 1.1 plus capture rate of the exact true model,

absolute bias in parameter estimation, and running time; use a separate test set to compute prediction accuracy. Note that the test set was generated in Step 1, and set aside for later use by inflating the target sample size $n$.

**Step 4**: Repeat the previous steps 1000 times and aggregate the results.

SCAD was implemented using the **ncvreg** package in R and $\gamma$ was fixed at 3.7, MC+ was implemented using the **plus** package. Adaptive lasso was implemented using the **glmnet** package and the initial weights are the inverse of absolute value of lasso estimates. For a fair comparison, GIC was used to select $\lambda$ in all algorithms. The ProSGPV algorithm was implemented using the **ProSGPV** package. The R code to replicate simulation results can be found at https://github.com/zuoyi93/r-code-prosgpv-linear.

Let $p$ denote the total number of variables, $s$ denote the number of true signals and $p-s$ denote the number of true noise variables. Suppose an algorithm identifies $r$ features as signals, of which only $v \leq \min(s,r)$ are correctly identified as signals. The remaining $r-v$ signals, if any, are mistakes. Some of the $p-r$ features that were called noise may have been signals too. Below are the formal definitions.

- Type I error rate (proportion of noise variables that were claimed to be signals) or
  $P(\text{Algorithm claims feature is a signal}|\text{feature is noise}) = (r-v)/(p-s)$
- Power (proportion of true signals that were correctly selected) or
  $P(\text{Algorithm claims feature is a signal}|\text{feature is signal}) = v/s$
- pFDR (proportion of algorithm identified signals that are really noise features) or
  $P(\text{feature is noise}|\text{Algorithm claims feature is a signal}) = (r-v)/r$
- pFNR (proportion of algorithm identified noise variables that are really true signals) or
  $P(\text{feature is signal}|\text{Algorithm claims feature is a noise}) = (s-v)/(p-r)$

### 1.4.2 Results and findings

We recorded whether or not each algorithm captured the exact true model in each iteration and compared the average capture rates over 1000 iterations in Figure 1.5. Power and Type I error rates are presented in

| | | **Truth** | | |
|---|---|---|---|---|
| | | **Signal** | **Noise** | **Subtotal** |
| **Algorithm** | **Signals** | v | r-v | r |
| | **Noise** | s-v | p-r-s+v | p-r |
| | **Subtotal** | s | p-s | p |

Table 1.1: Definitions of the metrics used in the simulation

Figure 1.6. False discovery proportions (pFDR) and false non-discovery proportions (pFNR) are presented in Figure 1.7. We also compared the mean absolute error (MAE) of all coefficient estimates, defined as $\frac{1}{p}\sum_{j=1}^{p}|\hat{\beta}_j - \beta_{0,j}|$, in Figure 1.8, where $\beta_{0,j}$ is the $j$th true coefficient. We compared the prediction accuracy of each algorithm, as measured by root mean square error (RMSE) in an independent test set in Figure 1.9. The effect of different parameter tuning methods on MC+ is illustrated in Figure 1.10. The comparison of computation time is shown in Figure 1.11.

In Figure 1.5, capture rates of the exact true model are compared under combinations of SNR and auto-correlation levels within the design matrix, when both $n > p$ and $n < p$.

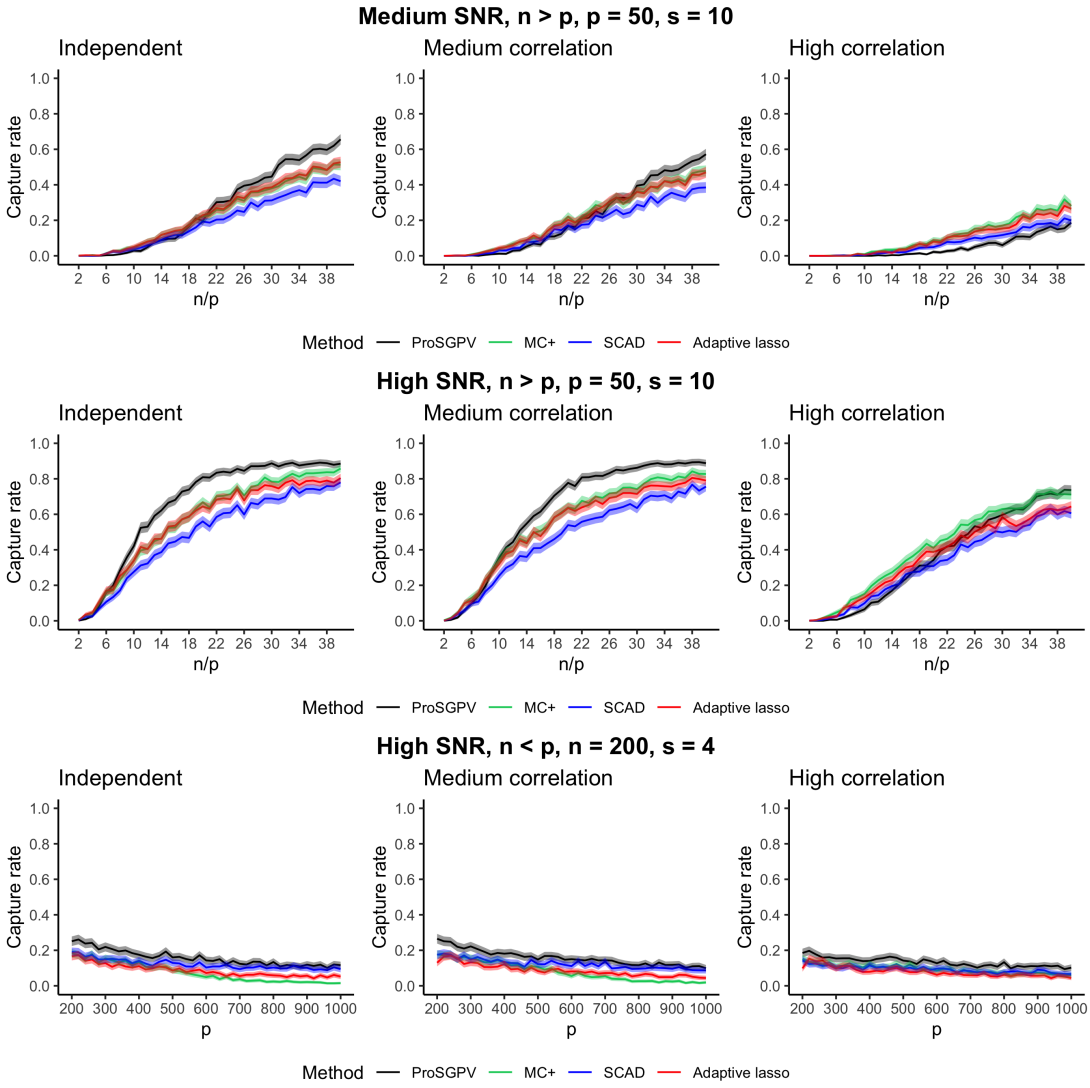


Figure 1.5: Capture rate of the exact true model under combinations of autocorrelation level, signal-noise-ratios, and $(n, p, s)$. In each panel, one algorithm has a colored solid line representing the average capture rate surrounded by the shaded 95% Wald interval over 1000 simulations.

When $n > p$, ProSGPV's support recovery rate increases as $n$ grows. It generally has the highest support

recovery rate except when the SNR is medium and correlation is high. MC+ and AL have similar capture rates, while SCAD is the worst among the four. When $p > n$, support recovery rates are low for all methods and decrease as $p$ increases in the data. ProSGPV again is the highest, followed by SCAD, AL, and MC+. We investigated factors driving the support recovery performance in Figure 1.6 and 1.7. When $n > p$, we see that all algorithms have decreasing Type I error rates, pFDR, pFNR, and increasing power. When $p > n$ and data are not highly correlated, GIC-based MC+ has notably higher pFDR than the others, indicating that it overfits the training data and includes many noise variables.

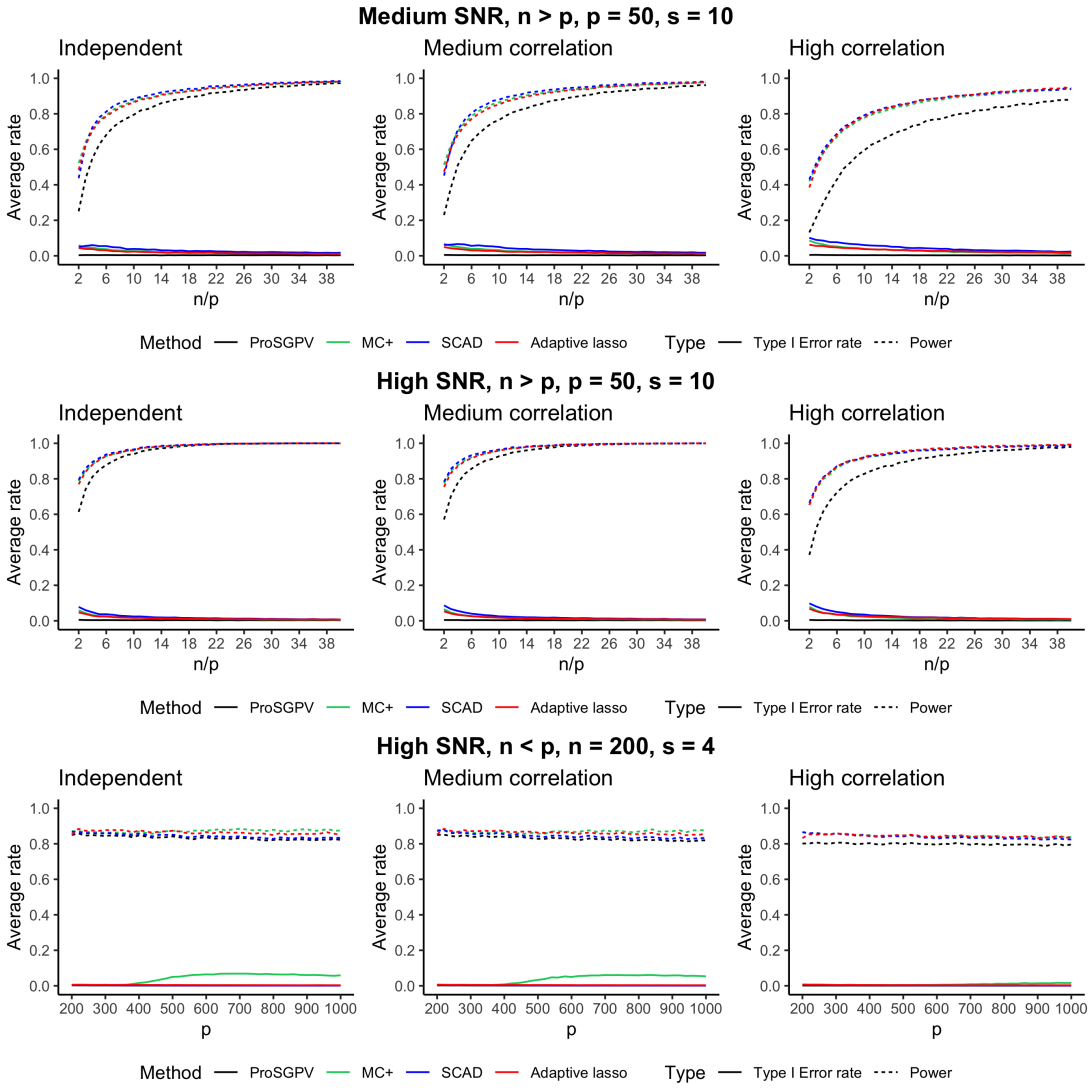


Figure 1.6: Estimated power and Type I error rates of all algorithms under combinations of autocorrelation level, signal-to-noise ratios, and $(n, p, s)$. Rates are plotted against either the ratio of the sample size $n$ over the number of variables $p$, or $p$ only.

Mean absolute error (MAE) is used to assess the parameter estimation error. When $n > p$, we used relative MAE which is defined as the ratio of an algorithm's MAE to that of the OLS model with only true

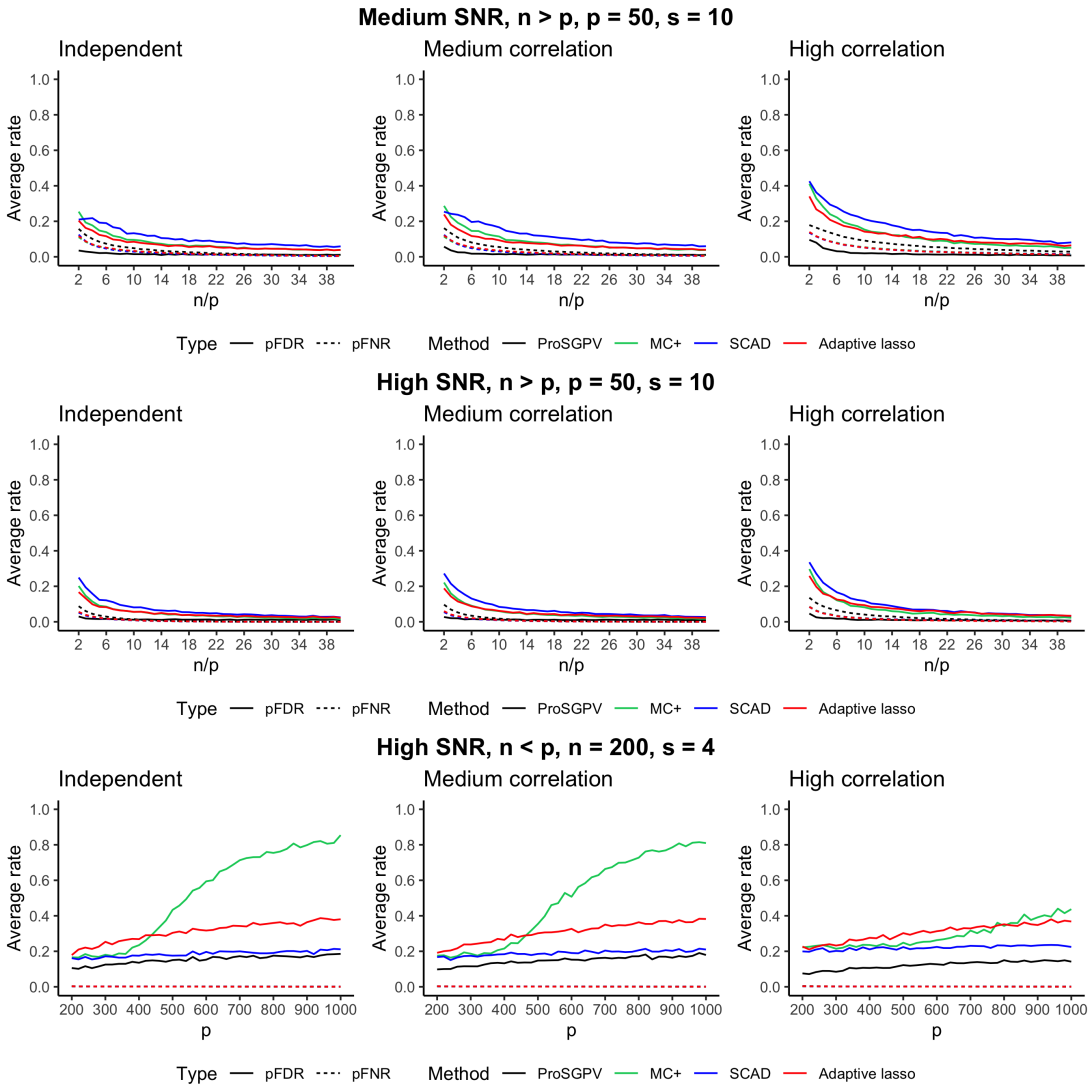


Figure 1.7: False discovery proportion (pFDR) and false non-discovery proportion (pFNR) of all algorithms under combinations of autocorrelation level, signal-to-noise ratios and $(n, p, s)$. Rates are plotted against either the ratio of sample size over the number of variables, or the number of variables only.

features. A good estimator would have an asymptotic relative MAE of one. When $p > n$, absolute MAE is used because no OLS fit is possible. Figure 1.8 displays the median (relative) MAE of four algorithms under various scenarios. The shading shows the first and third quartiles of the empirical (relative) MAE distribution.

In both $n > p$ and $n < p$ cases, ProSGPV has the lowest parameter estimation error. This should not be surprising for sparse settings with well-defined signals, as ProSGPV is effectively an $\ell_0$ penalization derivative and $\ell_0$ penalization drops small effects while keeping large ones. Johnson et al., 2015 showed that the parameter estimation risk of $\ell_0$-penalized regression can be infinitely better than that of the $\ell_1$-penalized regression under certain conditions and this is a practical example. The shape of the relative MAE from

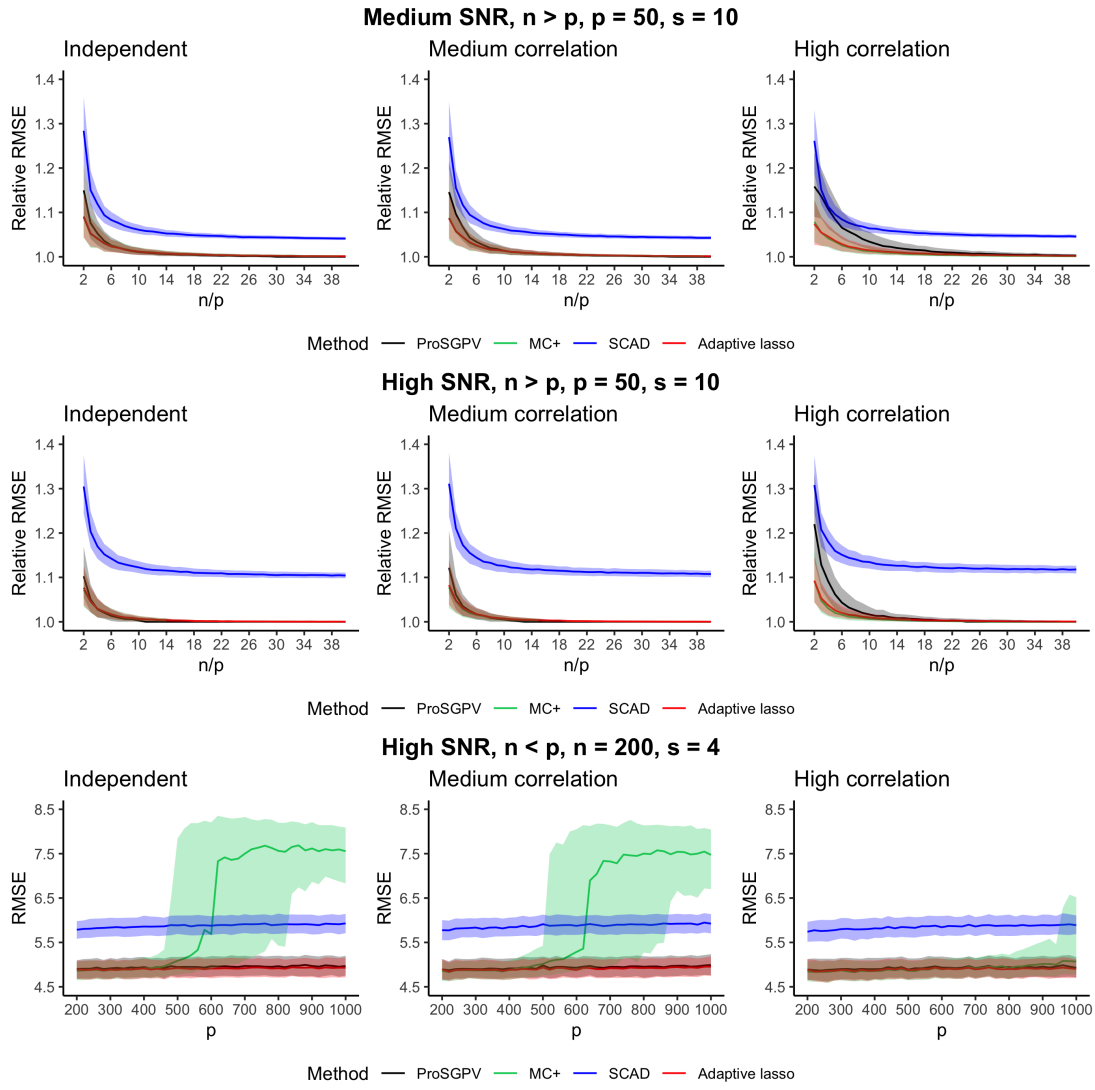


Figure 1.8: Parameter estimation error of all algorithms under combinations of autocorrelation level, signal-to-noise ratio, and $(n, p, s)$. In each panel, one algorithm has a colored solid line representing the median (relative) mean absolute errors surrounded by the shaded first and third quartiles over 1000 simulations.

ProSGPV generally follows what would be expected from a rate of $\sqrt{\log(n)/n}$. This rate matches the ideal rate of parameter estimation in the optimal model from any hard thresholding function, as suggested by Theorem 1 of Zheng et al., 2014. When $n > p$, AL and MC+ have very close performance and SCAD has the slowest rate of convergence. When $n < p$, the order stays the same for all except MC+. As $p$ passes 600, MC+ with GIC-based tunning selects more noise variables in the model and the parameter estimation performance is compromised. This can be remedied by using a universal $\lambda = \sigma\sqrt{(2/n)\log p}$ in MC+ (see Supplementary Figure 5). Fan and Tang, 2013 argued that MC+ has the same performance as SCAD when GIC is used. However, in their setting, $s$, $p$, and $n$ are allowed to grow together. In our case, $s$ is fixed at 4, $n$

is fixed at 200, and only $p$ grows.

In Figure 1.9, the prediction RMSE is calculated in an independent test set (40%) using models built with a training set (60%). Again, when $n > p$ the relative RMSE is used while when $n < p$ the absolute RMSE is used. Relative RMSE is defined as the ratio of the prediction RMSE from one algorithm to that from the OLS model with true signals only.

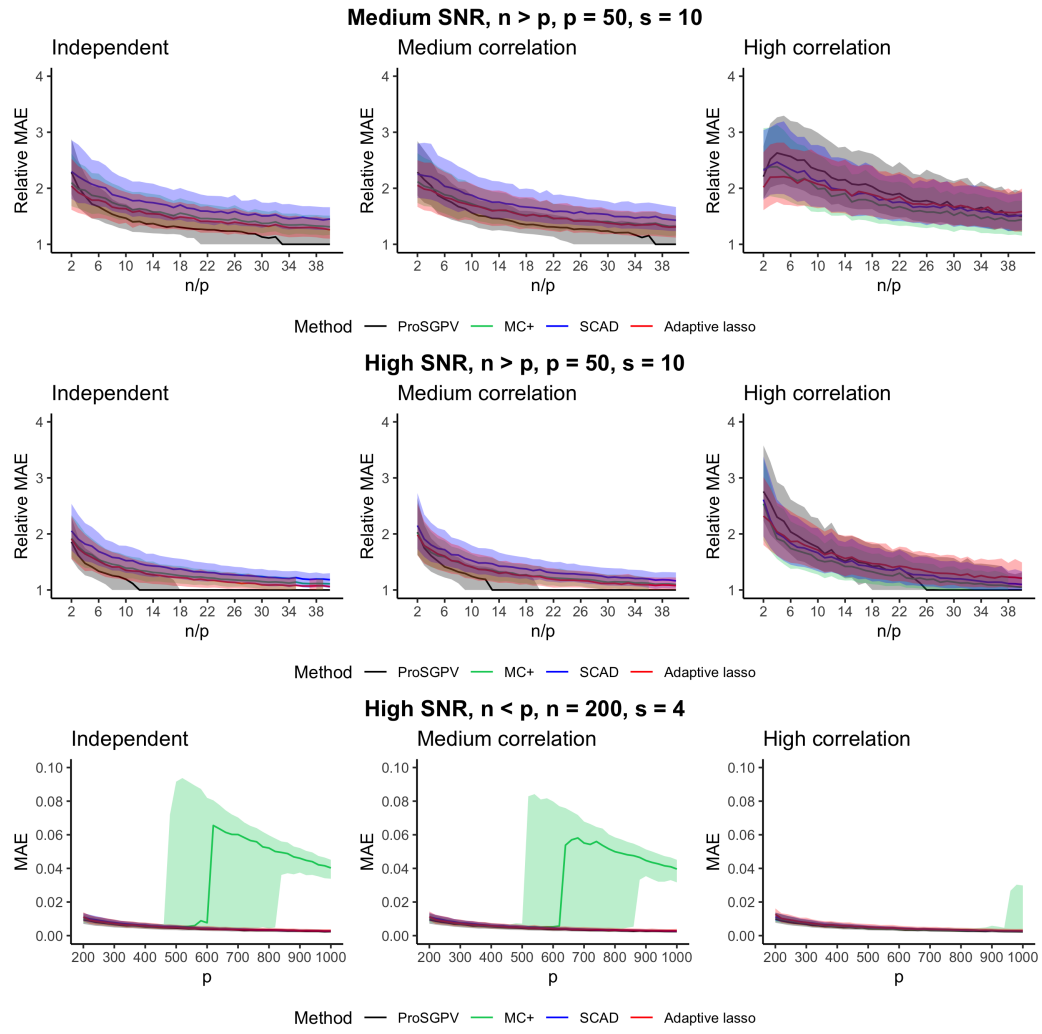


Figure 1.9: Comparison of prediction accuracy of all algorithms under combinations of autocorrelation level, signal-to-noise ratio, and $(n, p, s)$. Median (relative) root mean square errors are surrounded by their first and third quartiles over 1000 simulations.

When $n > p$, all algorithms have worse prediction performance than the true OLS model unless $n$ is really large. But their prediction RMSEs converge to the true OLS RMSE from above as $n$ increases. While ProSGPV is not optimized for prediction tasks, its predictive ability quickly catches up with other algorithms when $n/p > 6$. When $n < p$, ProSGPV and AL have the best performance followed by SCAD. SCAD can have better prediction performance when $\lambda$ is selected by cross-validation. However, in that case, its support

recovery is worse than that from the GIC-based SCAD. MC+ has much higher prediction error than the others when $p > 600$. That is because $\lambda$ selected by GIC leads to a dense model which includes many noise variables. The overfitted model has poor prediction performance in an external data set. However, this can be remedied by using a universal $\lambda$ in MC+, as shown in Figure 1.10.

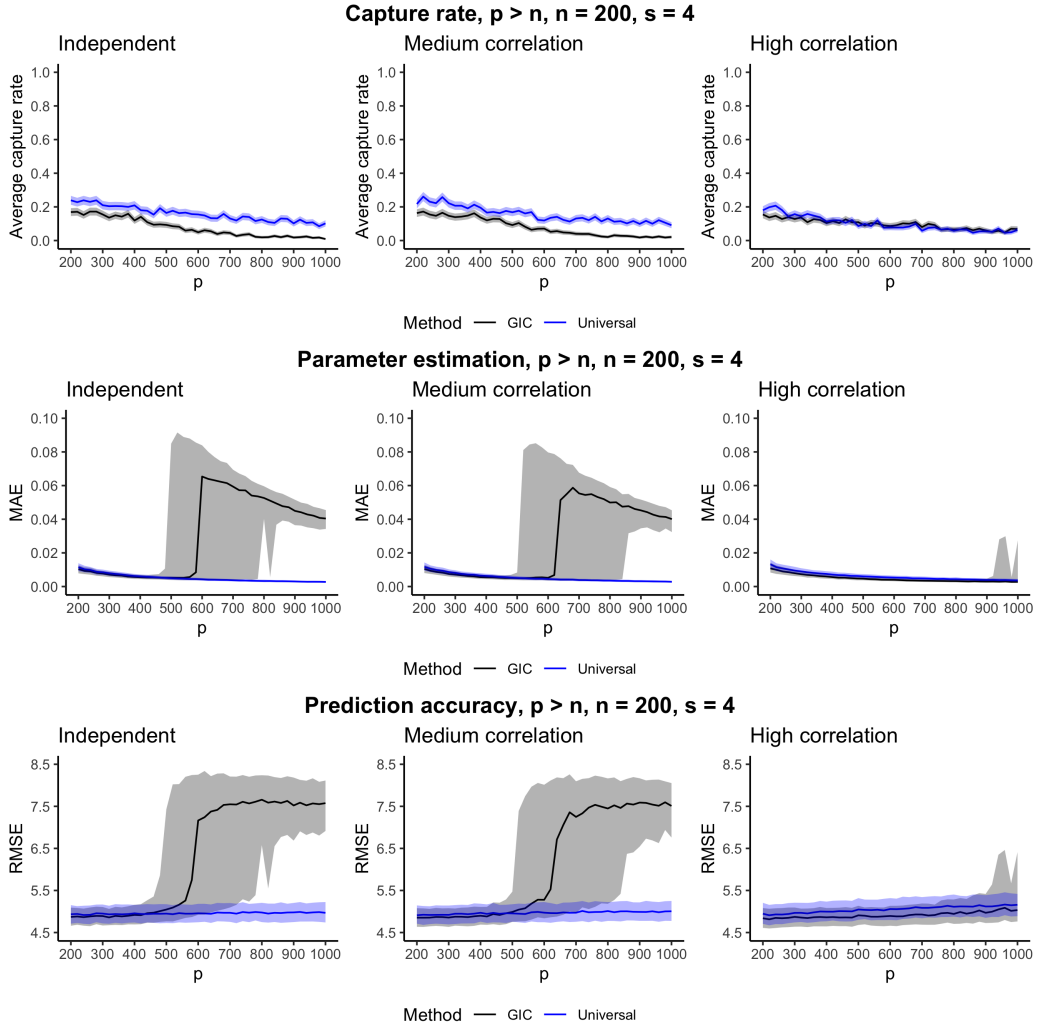


Figure 1.10: How different parameter tuning methods affects MC+. Support recovery rates, parameter estimation error measured by mean absolute error, and prediction accuracy measured by prediction root mean square error in an independent test set are compared for MC+ implemented in two ways. One way is to select the $\lambda$ that minimizes the generalized information criterion. The other way is to use a universal $\lambda = \sigma\sqrt{(2/n)\log p}$.

In Figure 1.11, the running time in seconds from all algorithms are compared. The computing environment was 2.6 GHz Dual-Core Intel Core i7 processor and 32 GB memory. ProSGPV and AL have the shortest computation time, followed by SCAD. MC+ is more time-consuming when data are highly correlated, or when $n < p$.

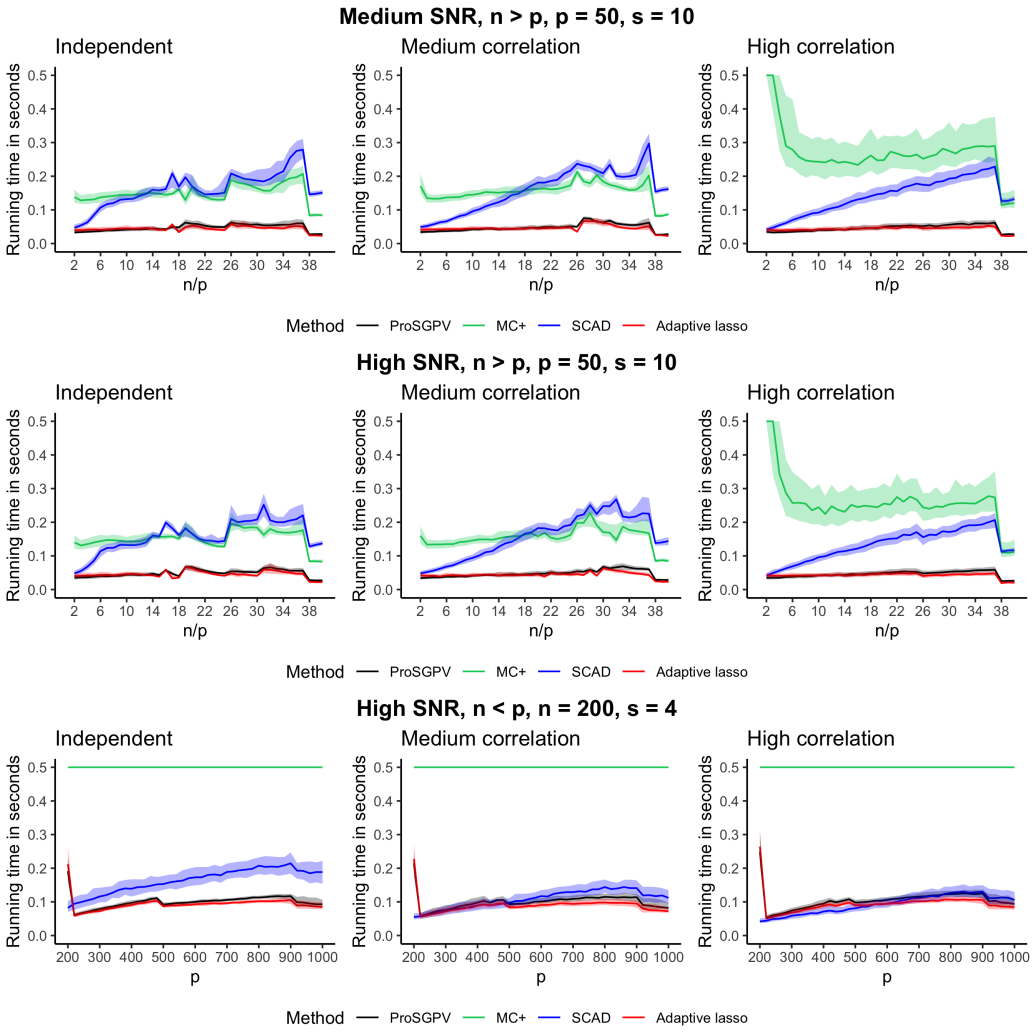


Figure 1.11: Comparison of computation costs of all algorithms under combinations of autocorrelation level, signal-to-noise ratios, and $(n, p, s)$. Solid lines are the median running time and the shades are first and third quartiles of the running time. For aesthetic reasons, values are censored at 0.5 second.

### 1.5 Real-world example

We illustrate our approach using the Tehran housing data (Rafiei and Adeli, 2016), which was high SNR ($R^2 = 0.98$) in the OLS model with all variables. We also explored the medium SNR case by removing potentially redundant variables until $R^2 = 0.4$. The Tehran housing data are available as a data object t.housing in the **ProSGPV** package. The data set contains 26 features and 372 records (see Table 1.2 for the variable description). The goal is to predict the sale price (variable 9 or V9). The explanatory variables consist of seven project physical and financial variables, 19 economic variables, all at baseline. Clustering and correlation patterns are displayed in Figure 1.12. We see that several explanatory variables form prominent clusters and that there is high pairwise correlation among the features. In particular, the price per square meter of the

unit at the beginning of the project (V8) has high correlation ($\rho = 0.98$) with the sale price (V9).

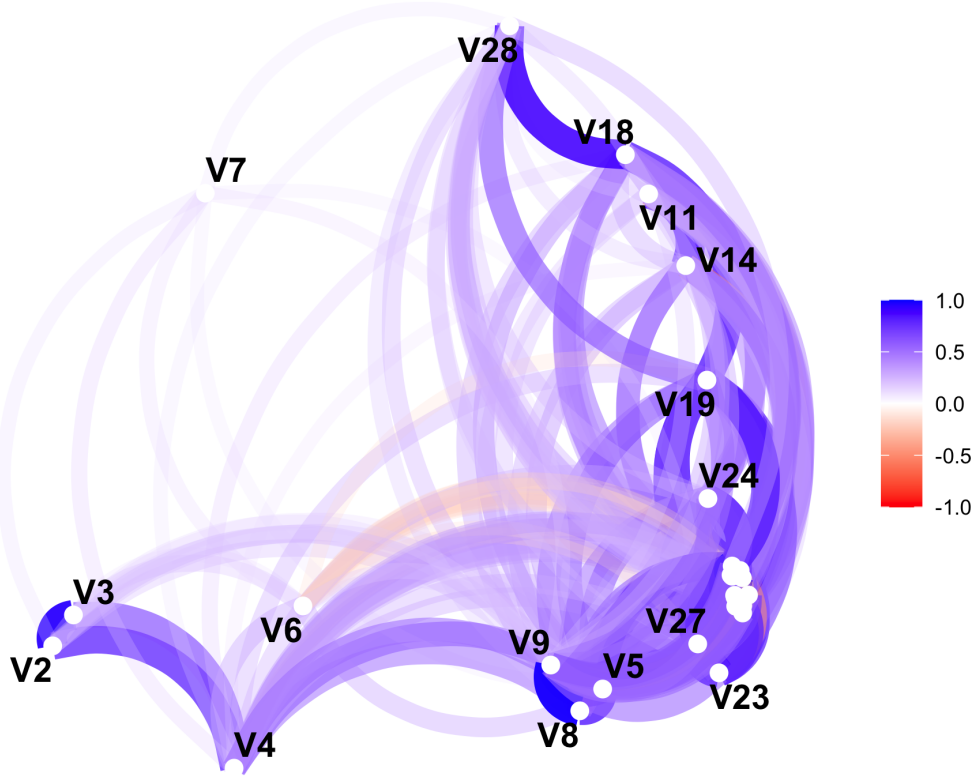


Figure 1.12: Clustering and correlation patterns of the Tehran housing data. The location of variables represents their clustering pattern. Each pair of variables is connected with a curve whose color represents the correlation level: strong positive correlation is in purple while strong negative correlation is in red.

We repeatedly split the data into a training set (70%) and a test set (30%). We applied AL, SCAD, MC+, and ProSGPV algorithms on the training set ($n$=260) with all the covariates. Prediction RMSE was calculated on the test set ($n$=112). We summarized the sparsity of the solutions (Figure 1.13) and prediction accuracy (Figure 1.14) over 1000 training-test split repetitions. SCAD overfits the training data and has the largest selection set. AL yields the sparsest model followed by ProSGPV. GIC-based MC+ yields a constant model size. Regarding the prediction performance, ProSGPV has the lowest median prediction error closely followed by AL and MC+, while SCAD has the largest test error because of overfitting. All algorithms select duration of construction (V7) and initial price per square meter (V8) with high frequency, and there is no consensus as for which other variables to include because of high correlation and clustering.

| Type | Variable name | Variable label |
|---|---|---|
| Outcome | V9 | Actual sales price |
| Project physical and financial features | V2 | Total floor area of the building |
| | V3 | Lot area |
| | V4 | Total Preliminary estimated construction cost based on the prices at the beginning of the project |
| | V5 | Preliminary estimated construction cost based on the prices at the beginning of the project |
| | V6 | Equivalent preliminary estimated construction cost based on the prices at the beginning of the project in a selected base year |
| | V7 | Duration of construction |
| | V8 | Price of the unit at the beginning of the project per square meter |
| Economic variables and indices | V11 | The number of building permits issued |
| | V12 | Building services index for preselected base year |
| | V13 | Wholesale price index of building materials for the base year |
| | V14 | Total floor areas of building permits issued by the city/municipality |
| | V15 | Cumulative liquidity |
| | V16 | Private sector investment in new buildings |
| | V17 | Land price index for the base year |
| | V18 | The number of loans extended by banks in a time resolution |
| | V19 | The amount of loans extended by banks in a time resolution |
| | V20 | The interest rate for loan in a time resolution |
| | V21 | The average construction cost by private sector at the completion of construction |
| | V22 | The average cost of buildings by private sector at the beginning of construction |
| | V23 | Official exchange rate with respect to dollars |
| | V24 | Nonofficial (street market) exchange rate with respect to dollars |
| | V25 | Consumer price index (CPI) in the base year |
| | V26 | CPI of housing, water, fuel & power in the base year |
| | V27 | Stock market index |
| | V28 | Population of the city |
| | V29 | Gold price per ounce |

Table 1.2: Variables in the Tehran housing data

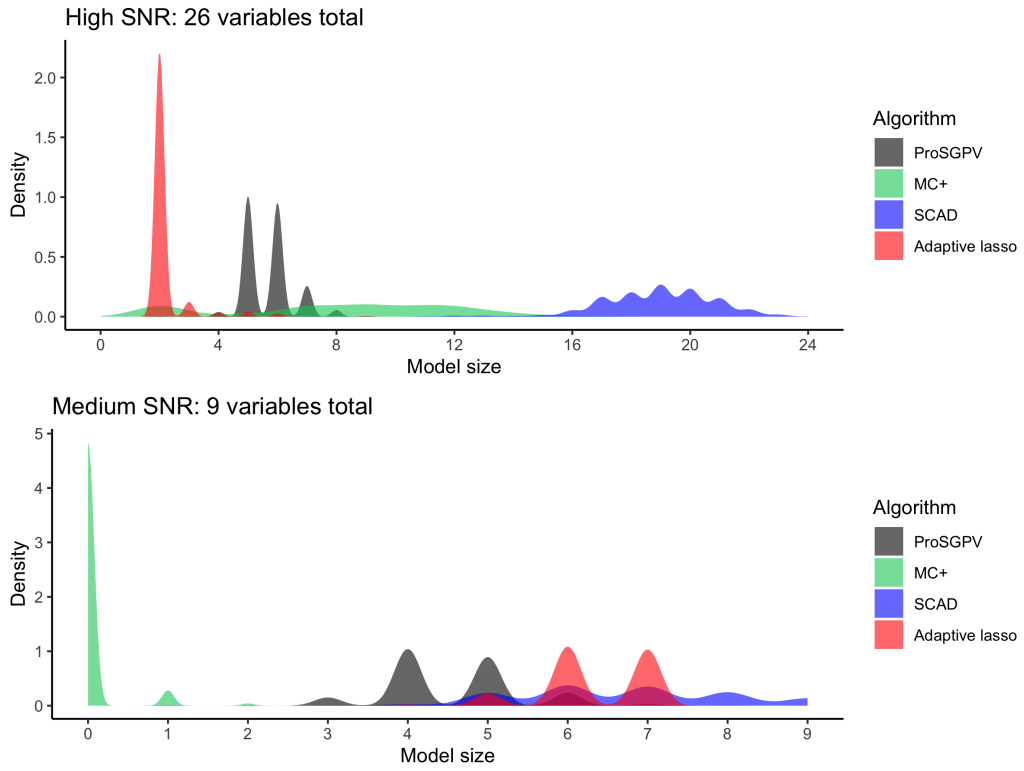


Figure 1.13: Histograms of model size from each algorithm in the training set over 1000 repetitions. The high signal-to-noise ratio case includes all 26 variables in the Tehran housing data, where the $R^2 = 0.98$ in the full ordinary least squares model; the medium signal-to-noise ratio case includes only nine variables, where the $R^2 = 0.41$ in the reduced ordinary least squares model.

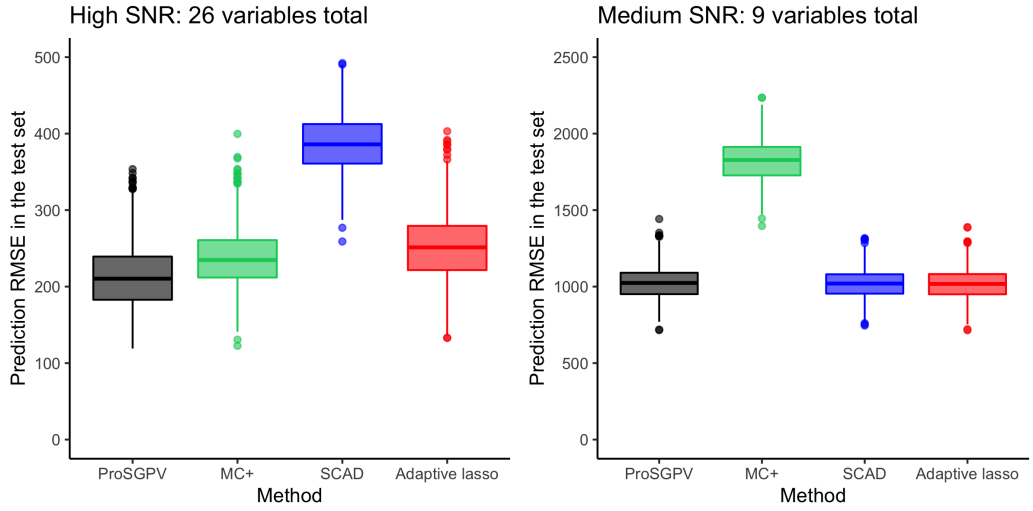


Figure 1.14: Boxplots of prediction root mean square errors from each algorithm in the test set over 1000 repetitions. The high signal-to-noise ratio case includes all 26 variables in the Tehran housing data, where the $R^2 = 0.98$ in the full ordinary least squares model; the medium signal-to-noise ratio case includes only nine variables, where the $R^2 = 0.41$ in the reduced ordinary least squares model.

To refine the analysis, we removed variables that had an absolute correlation with the outcome of 0.45 or greater. The remaining covariates explain 40% of the variability in the response, which represents medium SNR. Of the remaining nine variables, MC+ always selects zero variables. ProSGPV selects four or five variables with high frequency. AL selects six or seven variables with high frequency. SCAD selects more variables than ProSGPV and AL. ProSGPV, SCAD and AL have similar prediction performance, while MC+ has worse performance, due to the null model it selects. The common selected variables include total floor area of the building (V2), lot area (V3), the price per square meter of the unit at the beginning of the project (V8), and the number of building permits issued (V11). The R code to replicate the results is available at https://github.com/zuoyi93/r-code-prosgpv-linear.

### 1.6 Practical implications, limitations, and comments

A naïve way to perform variable selection is to screen variables by p-values. Such methods include forward selection, backward selection, and stepwise selection (Efroymson, 1966). However, these methods have serious drawbacks. They have poor capture rates of the true underlying model (Wang, 2009; Kozbur, 2018) and larger effective degrees of freedom (Kaufman and Rosset, 2014; Janson et al., 2015). In addition, the standard errors of the coefficient estimates are too small, which leads to over-optimistic discoveries (Harrell Jr, 2015). Better approaches do exist, but they are more complex, require specialized software, and are not fully adopted in routine applied practice. However, SGPVs offer a simple and effective option. It exhibits excellent statistical properties in both inference and prediction tasks without increased computation time.

Our simulation studies reinforce the notion that a model with good prediction ability does not necessarily lead to good inference. Comparing Figure 1.5 with Figure 1.9, we see that models optimized for prediction tend not to be optimized for inferential tasks, even when we use a different parameter tuning approach for each algorithm. This corroborates findings in the literature (Leng et al., 2006; Meinshausen et al., 2006; Wasserman and Roeder, 2009; Zheng et al., 2014; Giacobino et al., 2017; Shortreed and Ertefaie, 2017). This statement is important and bears repeating: models optimized for prediction tasks do not necessarily support good inference. Similar observations can be found by comparing the parameter estimation in Figure 1.8 with prediction performance in Figure 1.9. ProSGPV does a better job by yielding a model that is primed for inference and also has good prediction properties.

There is a link between SNR and the proportion of variance explained (PVE).

$$\text{PVE}(f) = 1 - \frac{\mathbb{E}(y - f(x)^2)}{\text{Var}(y)} = 1 - \frac{\text{Var}(\varepsilon)}{\text{Var}(y)} = \frac{\text{SNR}}{1 + \text{SNR}} \tag{1.5}$$

where $f$ is the mean function, and $x$ is independent from $\varepsilon$. Practically, when the $R^2$ is around 0.40 in the full

model, which is equivalent to a medium SNR in our simulation, ProSGPV has a comparable performance in support recovery and slightly better parameter estimation in large $n$; when the $R^2$ is above 0.66, which corresponds to a high SNR, ProSGPV has superior inference properties than the other algorithms.

There are some limitations. Sensitivity to tuning parameter specification is an issue for both implementation and generalizability of results. However, we found the findings to be fairly robust to tuning parameter specification. In results not shown here, we repeated the experiment using each algorithm's preferred method for choosing a tuning parameter and the general ordering of results remained stable. Another limitation is when the design matrix has high within-correlation, which is a challenging problem for any algorithm. Not unexpectedly, ProSGPV does not do well in support recovery and its parameter estimation and prediction performance suffers. We also note that the exact threshold function of the two-stage ProSGPV algorithm is difficult to conceptualize, as the null bound in the fully relaxed lasso has a different feature space than the full feature space. We are actively working on formulating solutions for the two-stage algorithm.

Despite these relatively minor limitations, the ProSGPV algorithm looks very promising. It gives up little in terms of prediction, and offers improved support recovery and parameter estimation properties compared to the class of standard procedures currently in use today. Also, the ProSGPV algorithm does not depend on tuning parameters that are hard to specify. It is fair to say that unlike traditional p-values, second-generation p-values can be used for variable selection and subsequent statistical inference.

# CHAPTER 2

## Variable Selection in Generalized Linear Models and Cox Models with Second-Generation P-Values

**Abstract**

Variable selection has become a pivotal choice in data analyses that impacts subsequent inference and prediction. In linear models, variable selection using Second-Generation P-Values (SGPV) has been shown to be as good as any other algorithm available to researchers. Here we extend the idea of Penalized Regression with Second-Generation P-Values (ProSGPV) to the generalized linear model (GLM) and Cox regression settings. The proposed ProSGPV extension is largely free of tuning parameters, adaptable to various regularization schemes and null bound specifications, and is computationally fast. Like in the linear case, it excels in support recovery and parameter estimation while maintaining strong prediction performance. The algorithm also preforms as well as its competitors in the high dimensional setting ($n > p$). Slight modifications of the algorithm improve its performance when data are highly correlated or when signals are dense. This work significantly strengthens the case for the ProSGPV approach to variable selection.

## 2.1 Introduction

Generalized linear models (GLM) (Nelder and Wedderburn, 1972) and the Cox proportional hazards model (Cox, 1972) are by now essential tools for regression modeling beyond linear models. GLMs can accommodate binary outcomes (disease status), count data (number of readmissions), and other type of continuous outcomes, while Cox models allow for time-to-event data (survival). Both are widely used in daily practice. While the magnitude of data – and their modeling options – has grown exponentially over the past decade, typically only a small finite set of features remains of interest to the researcher (Wainwright, 2009b; Loh and Wainwright, 2017; Gao et al., 2020). As such, variable selection remains a crucial task because it often leads to better risk assessment, parameter estimation and model interpretation (Shmueli et al., 2010). Sometimes, the number of variables, say $p$, can be (much) larger than the number of observations $n$ (e.g., see microarray, proteomic, and single nucleotide polymorphisms (SNPs) data). In those cases, penalized likelihood methods are often employed to encourage sparsity so that subsequent statistical inference is reliable.

In this paper, we extend the penalized regression with second-generation p-values (ProSGPV) (Zuo et al., 2021b) from the linear regression setting to GLMs and Cox models. The key ideas remain the same, although their implementation requires special attention outside of the linear case. First, the algorithm identifies a candidate set that is likely to contain the true signals. Then is proceeds to screen out noise variables using second-generation p-value (SGPV) methodology (Blume et al., 2018, 2019). Traditional $p$-values only measure the agreement between data and a point null hypothesis. By contrast, SGPVs measure the strength of an effect by comparing its entire uncertainty interval to a pre-specified interval null hypothesis. This idea of incorporating uncertainty into the variable selection procedure is novel and readily generalizable to different settings. We show below by simulation that the ProSGPV algorithm has very competitive support recovery, parameter estimation, and prediction performance in comparison to the current standards across Logistic, Poisson, and Cox models. Moreover, ProSGPV is largely insensitive to hard-to-interpret tuning parameters. The algorithm's framework has a flexible setup and is robust to modifications to many steps.

The rest of the paper is organized as follows. Section 2.2 describes the current landscape of pertinent variable selection methods for models of non-linear outcomes. Section 2.3 details the generalized ProSGPV variable selection algorithm. Section 2.4 presents simulation studies comparing inference and prediction performance of the aforementioned methods when $n > p$ and in high dimensional settings when $p > n$. Section 2.5 illustrates the ProSGPV algorithm using a real-world example. Section 2.6 summarizes key findings and discusses limitations.

## 2.2 Current landscape

Let $\{(X_i, Y_i)\}_{i=1}^n$ be a collection of $n$ independent observations, where $Y_i$ are independently observed response values given the $p$-dimensional predictor vector $X_i$. With a canonical link, the conditional distribution of $Y_i$ given $X_i$ belongs to an exponential family with density function $f(Y_i|\theta_i) \propto \exp(Y_i\theta_i - b(\theta_i))$, where $\theta_i = Z_i^T\beta$ with $Z_i = (1, X_i^T)^T$, $\beta = (\beta_0, \beta_1, ..., \beta_p)^T$, and $b(\theta)$ is assumed to be twice continuously differentiable with positive $b''(\theta)$. The loss function (negative log-likelihood) of a GLM model is

$$L(\beta) = -\frac{1}{n}\sum_{i=1}^{n}[Y_i Z_i^T\beta - b(Z_i^T\beta)] \tag{2.1}$$

Penalized likelihood methods optimize a loss function that is a slight variation on Equation (2.1), which is

$$L_\lambda(\beta) = -\frac{1}{n}\sum_{i=1}^{n}[Y_i Z_i^T\beta - b(Z_i^T\beta)] + p_\lambda(\beta) \tag{2.2}$$

where $p_\lambda(\beta)$ is the penalty function that involves a tuning parameter $\lambda$ and the coefficients $\beta$. In the case of a Cox model, the log partial likelihood is used for the likelihood and penalized by $p_\lambda(\beta)$. Almost all variable selection methods in this setting amount to selecting a particular penalty function.

### 2.2.1 Lasso

Lasso (Tibshirani, 1996, 1997) is one of the most widely used variable selection methods. It introduces an $\ell_1$ penalty to the loss function and shrinks effects towards zero. The sparsity induced by the $\ell_1$ penalty facilitates variable selection. The lasso GLM can be written as

$$L_\lambda^{\text{lasso}}(\beta) = -\frac{1}{n}\sum_{i=1}^{n}[Y_i Z_i^T\beta - b(Z_i^T\beta)] + \lambda\|\beta\|_1 \tag{2.3}$$

where $\|\cdot\|_1$ is the $\ell_1$-norm.

While parameter and risk estimation properties of lasso has been studied in high dimensional GLM under Lipschitz loss function(Meier et al., 2008) as well as usual quadratic loss in GLM (Wang et al., 2015), it is hard to find the ideal $\lambda$ that achieves excellent support recovery with excellent parameter

### 2.2.2 BeSS

Best subset selection (BSS) evaluates all sub-models in the feature space and searches for the one with the smallest loss criterion. Mathematically, BSS uses an $\ell_0$ penalty in the loss function. The BSS loss function can be written as

$$L_\lambda^{\text{BeSS}}(\beta) = -\frac{1}{n}\sum_{i=1}^{n}[Y_i Z_i^T\beta - b(Z_i^T\beta)] + \lambda\|\beta\|_0 \tag{2.4}$$

where $\|\cdot\|_0$ is the number of non-zero element in a vector. The BSS problem is nonconvex and NP-hard (Natarajan, 1995). A fast implementation of BSS for GLM/Cox models, called BeSS, was recently proposed Wen et al., 2020). When the golden section primal-dual active set (GPDAS) algorithm is used, there is no need to pre-specify the size of the model, and BeSS is still able to identify the best sub-model very quickly even when the number of features p is around 10,000. However, little is known about the asymptotic inference and prediction properties of BeSS solutions using GPDAS algorithm.

### 2.2.3 ISIS

Sure independence screening (SIS) was first proposed in linear regression (Fan and Lv, 2008), and later extended to GLM models (Fan and Song, 2010). SIS works by ranking the maximum marginal likelihood estimators in GLM/Cox - the univariate estimates - and then selecting the top variables for the final model. However, SIS can miss variables that are marginally and weakly correlated with the response, but highly important in a joint sense; SIS's dependence on marginal models may also cause it to rank some jointly unimportant variables too high. These and related issues were partially addressed by modifying the SIS algorithm, such as in multi-stage SIS, which is SIS followed by second-stage variable selection using lasso or smoothly clipped absolute deviation (SCAD), and iterative SIS (ISIS). The surprisingly simple ISIS often leads to good support recovery and parameter estimation. However, in order for ISIS to be variable selection consistent, the true signals should be sparse and not small (Fan and Song, 2010).

### 2.2.4 SGPV

Second-generation p-values (SGPV), denoted as $p_\delta$, were proposed for use in high dimensional multiple testing contexts (Blume et al., 2018, 2019). SGPVs attempt to resolve some of the deficiencies of traditional $p$-values by replacing the point null with a pre-specified interval null $H_0 = [-\delta, \delta]$. The interval null $H_0$ is the set of effects that are scientifically indistinguishable or immeasurable from the point null hypothesis, due to limited precision or practicality. The 'effects' are typically measured as log odds ratios (logistic regression), log rate ratios (Poisson regression), and log hazards ratios (Cox regression). SGPVs are loosely defined as the fraction of data-supported hypotheses that are also null, or nearly null, hypotheses. Formally, let $\theta$ be a parameter of interest, and let $I = [\theta_l, \theta_u]$ be the interval estimate of $\theta$ whose length is given by $|I| = \theta_u - \theta_l$. If we denote the length of the interval null by $|H_0|$, then the SGPV $p_\delta$ is defined as

$$p_\delta = \frac{|I \cap H_0|}{|I|} \times \max\left\{\frac{|I|}{2|H_0|}, 1\right\} \tag{2.5}$$

where $I \cap H_0$ is the intersection of two intervals. The correction term $\max\{|I|/(2|H_0|), 1\}$ fixes the problem when the interval estimate is very wide, i.e., when $|I| > 2|H_0|$. In that case, the data are effectively inconclusive and the main quantity $|I \cap H_0|/|I| = |H_0|/|I|$ does not properly reflect this inconclusive nature of the data. As such, SGPV indicate when data are compatible with null hypotheses ($p_\delta = 1$), or with alternative hypotheses ($p_\delta = 0$), or when data are inconclusive ($0 < p_\delta < 1$).

By design, SGPV emphasize effects that are clinically meaningful by exceeding a pre-specified null level. Empirical studies have shown that SGPV can identify feature importance in linear models (Blume et al., 2018, 2019; Zuo et al., 2021b), and we will show below that SGPV can also be used for variable selection in GLM/Cox models.

### 2.3 ProSGPV algorithm in GLM/Cox

The penalized regression with second-generation p-values (ProSGPV) algorithm was proposed for linear regression (Zuo et al., 2021b), but it is readily adapted to accommodate Logistic, Poisson, and Cox regressions.

#### 2.3.1 Steps

The steps of the ProSGPV algorithm in GLM/Cox models are shown below in Algorithm 2.

**Algorithm 2** The ProSGPV algorithm in GLM/Cox models

**procedure** PROSGPV($X$, $Y$)
  **Stage one**: Find a candidate set
    Fit a lasso and evaluate it at $\lambda_{\text{gic}}$
    Fit GLM/Cox models on the lasso active set
  **Stage two**: SGPV screening
    Extract the confidence intervals of all variables from the previous step
    Calculate the mean coefficient standard error $\overline{SE}$
    Calculate the SGPV for each variable where $I_j = \hat{\beta}_j \pm 1.96 \times SE_j$ and $H_0 = [-\overline{SE}, \overline{SE}]$
    Keep variables with SGPV of zero
    Refit the GLM/Cox with selected variables
**end procedure**

In the first stage, lasso is evaluated at the $\lambda$ that minimizes the generalized information criterion (Fan and Tang, 2013). The second step is a fully relaxed lasso (Meinshausen, 2007) on variables with non-zero coefficients in the first step. The second stage uses SGPVa to screen variables. Throughout this paper,

SGPV are calculated based on 95% confidence intervals, though other interval choices would also work. The ProSGPV algorithm has been implemented in the ProSGPV R package, which is available from the Comprehensive R Archive Network (CRAN) at https://CRAN.R-project.org/package=ProSGPV.

### 2.3.2 Flexibility of the algorithm

In ProSGPV, lasso is used in the first-stage to leverage its quick computation speed. Other dimension reduction methods, e.g. elastic net or adaptive lasso or even SIS, once properly tuned, should also lead to good inference and prediction performance in the ProSGPV algorithm as well (this tangent idea will be evaluated in a separate venue). Moreover, ProSGPV only requires that the choice of $\lambda$ for the first-stage lasso be only in a certain range. This is because the feature space for the fully relaxed lasso is discrete and so there exists a range of $\lambda$s that lead to the same fully relaxed lasso model (See Figure 2.1 for details). In practice we suggest using the $\lambda$ that minimizes the generalized information criterion. Figure 2.2 shows that as long as $\lambda$ falls into a reasonable range, the support recovery of ProSGPV is consistently good. Although using exactly $\lambda_{\text{gic}}$ may be preferable in high dimensional settings where $p > n$ (Fan and Tang, 2013).

Another flexible aspect of the ProSGPV algorithm is the choice of the null bound in the second stage. This can be thought of as a tuning parameter, but one that is less sensitive. For a default, we choose the average standard error (SE) of coefficient estimates. For example, in Logistic regression, the null bound would be set at the SE of log odds ratio. Other forms of the null bound may lead to decent support recovery as well, although the Type I/Type II error tradeoff needs to be formally assessed. In Figure 2.3, we compared the performance of ProSGPV using various null bounds such as $\overline{SE}$, $\overline{SE} * \sqrt{\log(n/p)}$, $\overline{SE}/\sqrt{\log(n/p)}$, $\overline{SE} * \sqrt{n/p}/2$, and the constant zero, in logistic regression. $\overline{SE} * \sqrt{n/p}/2$ is roughly constant when $n$ increases with fixed $p$ and is the largest among all bounds. When the null bound is multiplied by $\sqrt{\log(n/p)}$, or divided by $\sqrt{\log(n/p)}$, the bound is inflated or deflated at each $n$. When null bound is zero, it's effectively selecting variables using the traditional $p$-values. The original null bound achieves variable selection consistency at the fastest rate. The other null bounds either are not variable selection consistent when $n$ is large (0 and $\overline{SE}/\sqrt{\log(n/p)}$), or achieve the consistency at a slower rate ($\overline{SE} * \sqrt{\log(n/p)}$ and $\overline{SE} * \sqrt{n/p}/2$). Hence our recommendation for using the original null bound.

An important note is that logistic regression fitted by maximum likelihood can suffer from inflated parameter estimation bias due to complete/quasi-complete separation when n is small (Albert and Anderson, 1984; Rahman and Sultana, 2017). A Jeffreys-prior penalty (Kosmidis and Firth, 2021) was included in ProSGPV to address this issue. A comparison of the two models is shown in the Figure 2.4. When signals are dense and $n$ is small, Logistic regression with Jeffreys prior has lower parameter estimation bias and better prediction accuracy than Logistic regression fitted by maximum likelihood.

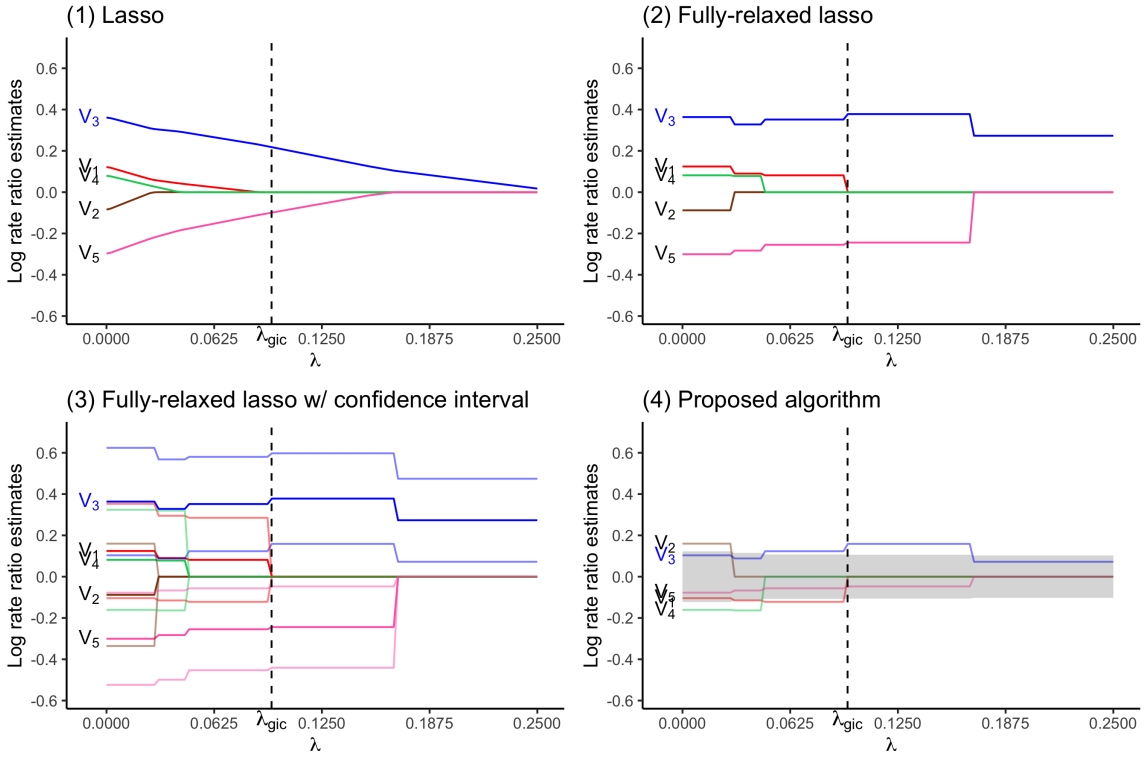


Figure 2.1: How ProSGPV works in a Poisson regression. The true data generating model contains only $V_3$. (1) presents the lasso solution path. (2) shows the fully relaxed lasso path. (3) shows the fully relaxed lasso paths with their 95% confidence intervals (in lighter color). (4) illustrates the ProSGPV selection path. The shaded area is the null region; the colored lines are each 95% confidence bound that is closer to the null region.

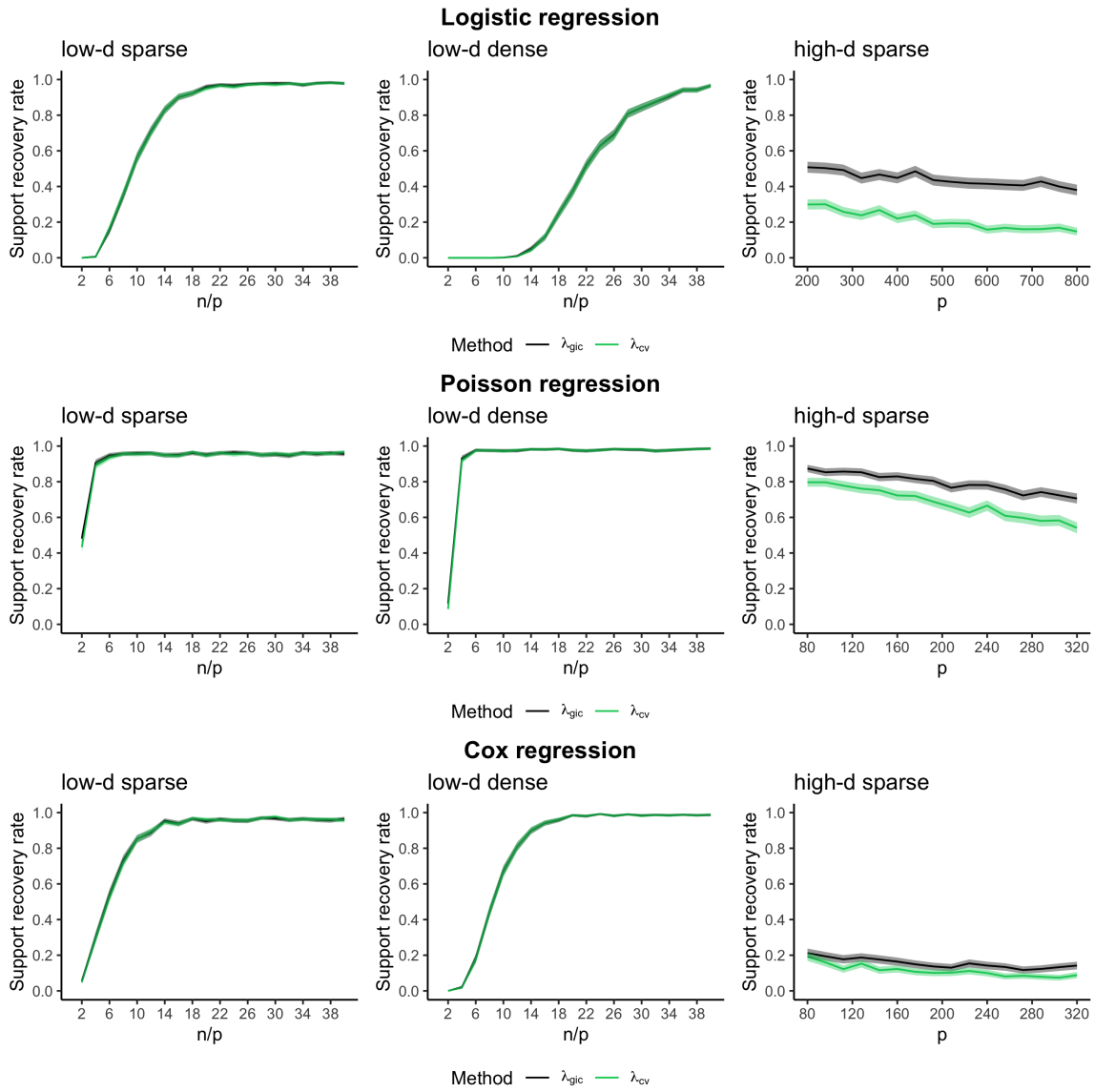


Figure 2.2: Sensitivity of the $\lambda$ in the first stage of ProSGPV. Mean capture rates of the exact true model are surrounded by 95% Wald confidence intervals over 1000 simulations. Black is the ProSGPV algorithm implemented with $\lambda_{\text{gic}}$, which is the $\lambda$ that minimizes the generalized information criterion. Green is the ProSGPV algorithm implemented with $\lambda$ from a uniform distribution of $(0.8\lambda_{\text{min}}, 1.2\lambda_{\text{1se}})$, where $\lambda_{\text{min}}$ is the $\lambda$ that minimizes the cross validation error, and $\lambda_{\text{1se}}$ is the largest $\lambda$ that yields a cross validation error that is within one standard error of the minimal cross validation error.

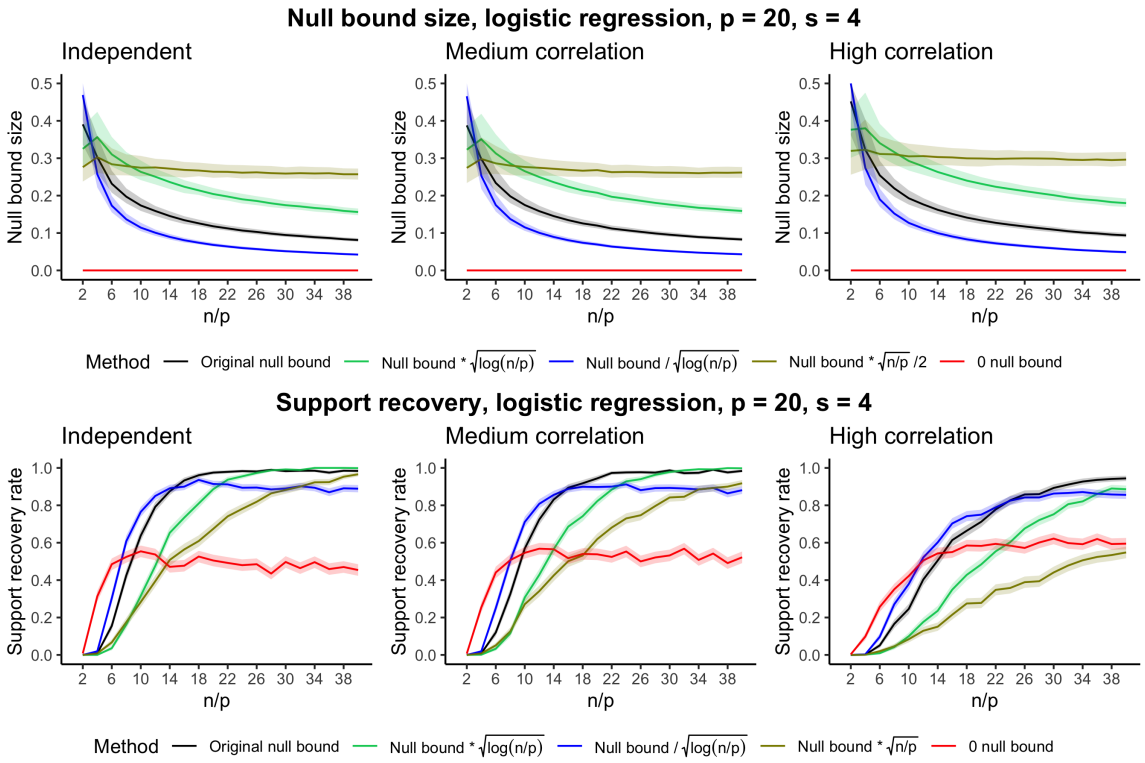


Figure 2.3: Sensitivity of the null bound in ProSGPV. Different null bounds and their corresponding support recovery rates are compared in Logistic regression at three correlation levels. Medians are surrounded with first and third quartiles from 1000 simulations. Null bound choices include the original null bound ($\overline{SE}$), $\overline{SE} * \sqrt{\log(n/p)}$, $\overline{SE} / \sqrt{\log(n/p)}$,$\overline{SE} * \sqrt{n/p}/2$, and 0.

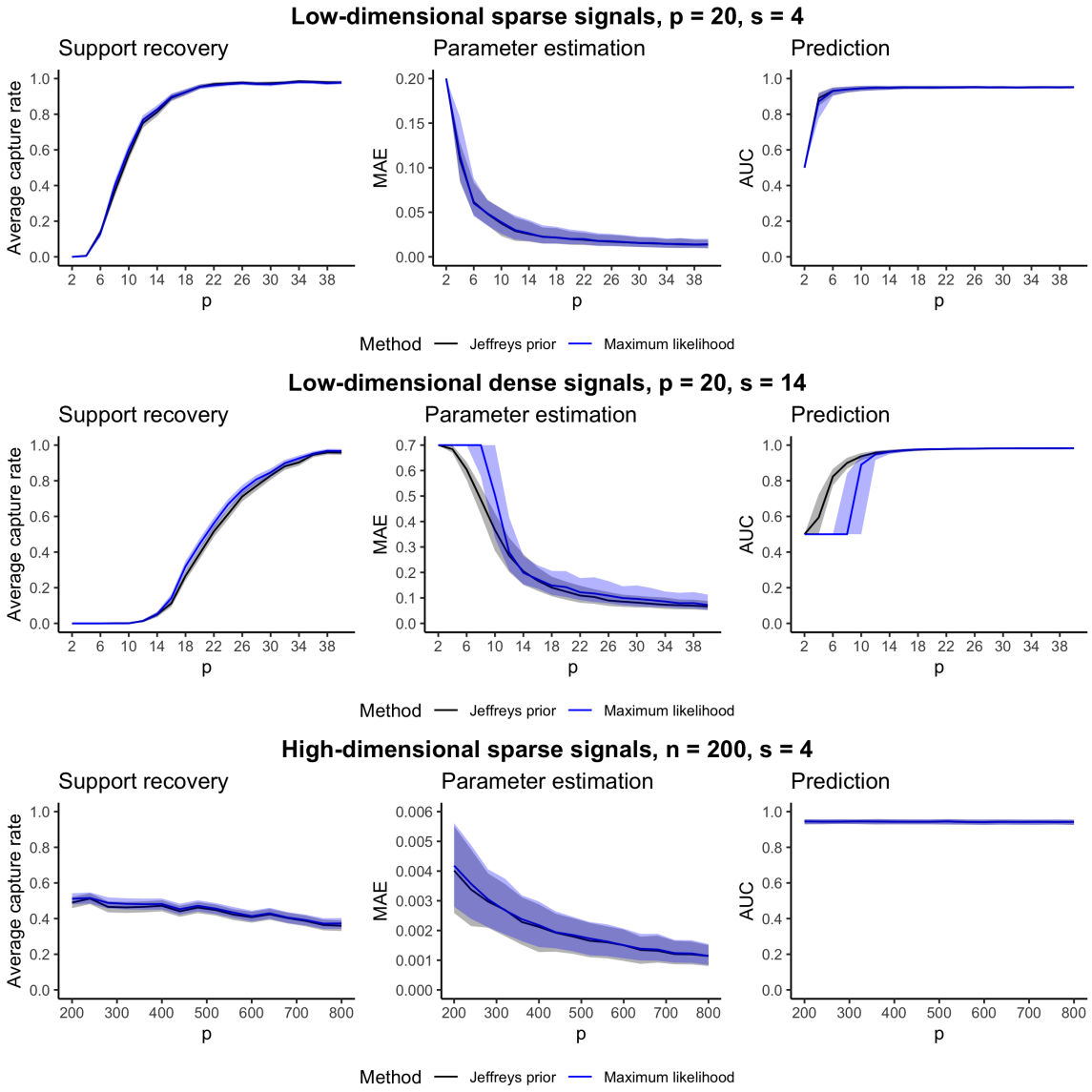


Figure 2.4: Comparison of maximum likelihood fitted and Jeffreys prior penalized logistic regressions. Average capture rates of the exact true model, average mean absolute errors (MAE), and average prediction area under the curve (AUC) in a separate test set are compared over 1000 simulations. For capture rates, means are surrounded by 95% Wald confidence intervals. For MAE and prediction AUC, medians are surrounded by first and third quartiles from the simulation.

### 2.3.3 Solution

The solution to the ProSGPV algorithm $\hat{\beta}^{pro}$ is

$$\begin{gathered} \hat{\beta}^{\text{pro}} = \hat{\beta}^{\text{gc}}_{|S} \in \mathbb{R}^p, \text{ where} \\ S = \{k \in C : |\hat{\beta}^{\text{gc}}_k| > \lambda_k\}, C = \{j \in \{1,2,...,p\} : |\hat{\beta}^{\text{lasso}}_j| > 0\} \end{gathered} \tag{2.6}$$

where $\hat{\beta}^{\text{gc}}_{|S}$ is a vector of length $p$ with non-zero elements being the GLM/Cox coefficient estimates from the model with variables only in the set $S$. $S$ is the final selection set and $C$ is the candidate set from the first-stage screening. The coefficients are log odds ratios in Logistic regression, log rate ratios in Poisson regression, and log hazard ratios in Cox regression. Note that the cutoff $\lambda_j = 1.96 * SE_j + \overline{SE}$.

Like the linear setting, ProSGPV can be seen as a hard thresholding function. In the saturated GLM/Cox models one can often order coefficients such that true signals all have greater coefficient estimates than noise variables do. The cutoff $\lambda_j$ aims to separate signal variables from noise variables. The only routine exception to this ordering is when the data have weak signals or high correlation, and in this case, no algorithms can fully recover the true support. (Zhao and Yu, 2006; Wainwright, 2009a)

### 2.3.4 Example

An example of how ProSGPV works in Poisson regression is shown below. There are five explanatory variables $(V_1, .., V_5) \in \mathbb{R}^{100\times 5}$ and the covariance matrix is autoregressive with autocorrelation being 0.5 and standard deviation of 1. The response $Y \in \mathbb{R}^{100\times 1}$ is simulated from Poisson distribution with mean of $\exp(\beta_3 V_3)$ where $\beta_3 = 0.25$. The goal is support recovery, i.e. to discover which variable is truly associated with the outcome. Figure 2.1 shows how ProSGPV succeeds while lasso and fully relaxed lasso fail at $\lambda = \lambda_{\text{gic}}$ by selecting $V_3$ and $V_5$.

## 2.4 Simulation studies

Extensive simulation studies were conducted to evaluate inference and prediction performance of the ProSGPV algorithm compared to existing standard methods, in both traditional $n > p$ and high-dimensional $p > n$ settings. The models under consideration include Logistic regression, Poisson regression, and Cox proportional hazards regression.

### 2.4.1 Design

Given sample size $n$, dimension of explanatory variables $p$, sparsity level $s$, true coefficient vector $\beta_0 \in \mathbb{R}^p$, autocorrelation level $\rho$ and standard deviation $\sigma$ in the covariance matrix, the simulation steps are as follows.

Step 1: Draw $n$ rows of the input matrix $X \in \mathbb{R}^{n\times p}$ i.i.d. from $N_p(0,\Sigma)$, where $\Sigma \in \mathbb{R}^{p\times p}$ has entry $(i,j)$ equal to $\sigma^2\rho^{|i-j|}$

Step 2: For Logistic regression, generate the linear vector $z = X\beta_0 \in \mathbb{R}^{n\times 1}$, where the intercept is 0, and generate response $Y \in \mathbb{R}^{n\times 1}$ from Bernoulli distribution with probability $pr = 1/(1+\exp(-z))$ For Poisson regression, generate $z \in \mathbb{R}^{n\times 1}$ as before, and generate response $Y \in \mathbb{R}^{n\times 1}$ from Poisson distribution with mean $\exp(z)$. For Cox regression, generate time to event from Weibull distribution with scale $\lambda_{\text{weibull}} = 2$, shape $k = 1$, and mean $\exp(z)$. generate censoring time from Exponential distribution with rate $\tau = 0.2$

Step 3: Run ProSGPV, lasso, BeSS, and ISIS, and record whether each algorithm recovers exactly, and only, the true underlying features; compute the proportion of correctly captured true features (power) and the proportion of incorrectly captured noise features (Type I Error rate); compute the false discovery proportion (pFDR) and false non-discovery rate (pFNDR); compute the mean absolute bias in parameter estimation and prediction area under the curve (AUC) in a separate test set except for Cox model

Step 4: Repeat the previous steps 1000 times and aggregate the results over iterations

$\beta_0$ has $s$ non-zero values equally spaced between $\beta_l$ and $\beta_u$, at random positions, and the remaining coefficients are set to zero. The non-zero coefficients are half positive and half negative. $\rho$ is 0.35 and $\sigma$ is 2. Additional simulation parameters are summarized in Table 2.1.

The ProSGPV algorithm was implemented using the ProSGPV package; lasso was implemented using glmnet package and was evaluated at $\lambda_{\text{min}}$; BeSS was implemented using BeSS package; and ISIS was implemented using SIS package. The code to replicate all results in this paper is available at https://github.com/zuoyi93/r-code-prosgpv-glm-cox.

### 2.4.2 Results and findings

The capture rates of the exact true model are shown in Figure 2.5. The average coefficient estimation error is compared in Figure 2.6. The prediction accuracy is displayed in Figure 2.7. Power (the fraction of true features that were captured) and Type I Error rate are compared in Figure 2.8. False discovery proportion (pFDR) and false non-discovery proportion (pFNR) are compared in Figure 2.9. Computation time is shown in Figure 2.10.

In Figure 2.5, capture rates of the exact true model are compared under Logistic regression, Poisson regression, and Cox regression with various $(n,p,s)$ combinations. Overall, ProSGPV has the highest capture rate in almost all scenarios, particularly in high-dimensional setting where $p > n$. But there are exceptions. For example, ProSGPV has a low capture rate in logistic regression when signals are dense and $n/p$ is small (low-d dense column in Figure 2.5). In that setting, BeSS has the best support recovery among the four, with ISIS not far behind. Interestingly, ISIS recovers support well in certain low-dimensional models but fails in

high dimensional settings. Here, Lasso is evaluated at $\lambda_{\min}$ which is optimized for prediction and completely fails in the support recovery task.

We further investigated the support recovery results by comparing their Type I error rates and power in Figure 2.6 and false discovery proportion (pFDR) and false non-discovery proportion (pFNR) in Figure 2.7. In general, ProSGPV selects most signal variables and seldom includes noise variables. However, ProSGPV has a low power in a dense Logistic regression when $n$ is small. BeSS selects more signals than ProSGPV at the cost of including more noise variables. ISIS selects slightly fewer signal variables and more noise variables, which becomes problematic for high-dimensional support recovery. Interestingly, lasso fails the support recovery task because it selects a large model that, while including almost all signal variables, has numerous noise variables. We see that ProSGPV has superior or comparable support recovery performance as the other standard procedures with the exception of the low-dimensional dense signal logistic regression. This happens in part because, in that case, the null bound is slightly higher than the noise level and ProSGPV includes too many noise variables. This can be fixed by replacing the constant null bound with an adjusted null bound based on the generalized variance inflation factor (GVIF) (Fox and Monette, 1992). Each coefficient standard error is inversely weighted by GVIF and then summed to derive an adjusted null bound. GVIF-adjusted ProSGPV has improved inference and prediction performance than original ProSGPV when signals are dense, or correlation is high in the design matrix. The improvement is illustrated in Figure 2.11 and this bound is an option in the ProSGPV package.

In Figure 2.6, parameter estimation bias, as measured by mean absolute error (MAE), is compared under same scenarios in Figure 2.5. ProSGPV generally has the lowest MAE in all cases. BeSS has low bias in low-dimensional case but has high variance in parameter estimation in high-dimensional Logistic and Cox regressions. ISIS and lasso tend to have decent parameter estimation only when signals are sparse.

In Figure 2.7, prediction accuracy, as measured by average area under the curve (AUC) in Logistic regression and root mean square error (RMSE) in Poisson regression, is compared under combinations of $(n, p, s)$ using an independent test set. In Logistic regression, when $n > p$, most algorithms reach a very high AUC as $n/p$ passes 10; when $p > n$, all algorithms achieve similar AUC even when $p$ grows. In Poisson regression, when $n > p$ and signals are sparse, all algorithms have very high prediction accuracy. When $n > p$ and signals are dense, ISIS has higher prediction error than the other algorithms. When $p > n$, ISIS and lasso has worse prediction than ProSGPV and BeSS.

Running time of all algorithms is compared in Figure 2.10. When $n > p$, lasso takes the longest time to run, likely due to the cross-validation step. ISIS usually takes the second longest time to run. BeSS can take longer to run in Cox regression when $n$ is large. ProSGPV is among the algorithms that have the shortest computation time. The spike in time when $n$ is small could be due to the convergence issue given a small

sample size. When $p > n$, ISIS and lasso are more time-consuming than BeSS and ProSGPV.

## 2.5 Real world data

We applied ProSGPV in a real-world data example to evaluate the sparsity of its solution and prediction accuracy. Lower back pain can be caused by a variety of problems with any parts of the complex, interconnected network of spinal muscles, nerves, bones, discs or tendons in the lumbar spine. Dr. Henrique da Mota collected 12 biomechanical attributes from 310 patients, of whom 100 are normal and 210 are abnormal (Disk Hernia or Spondylolisthesis). The goal is to differentiate the normal patients from the abnormal using those 12 variables. The biomechanical attributes include pelvic incidence, pelvic tilt, lumbar lordosis angle, sacral slope, pelvic radius, degree of spondylolisthesis, pelvic slope, direct tilt, thoracic slope cervical tilt, sacrum angle, and scoliosis slope. This spine data were acquired from University of California Irvine Machine Learning Repository (Dua and Graff, 2017) on March 4th, 2021 and is available in the ProSGPV package.

The clustering and correlation pattern of the data are shown in Figure 2.12. Most features are weakly associated with the outcome, and some of them are heavily correlated with each other. To understand the sparsity of solutions from various algorithms, we repeatedly split the data set into a training (70%) and test (30%) set. In each of 1000 repetitions, we ran ProSGPV, lasso, BeSS, and ISIS on the training set, recorded the model size, and calculated the prediction AUC in the test set. Training model sizes and most frequent models are summarized in Figure 2.13. Prediction AUC are summarized in Figure 2.14. Lasso selected the greatest number of variables while ProSGPV selected a sparser model on average. Pelvic radius and degree of spondylolisthesis were selected by three out of four algorithms. All algorithms yielded similar prediction performance on the test set. In summary, ProSGPV produced a sparse model with high predictive ability.

## 2.6 Discussion

### 2.6.1 When ProSGPV excels

As shown from simulation studies in Section 2.4, ProSGPV can recover the true support with high probability in classic $n > p$ and high dimensional $p > n$ settings. When $p > n$, signals are generally assumed to be sparse; otherwise, to our knowledge, no known algorithms would successfully recover the true support (Fan and Song, 2010; Wang et al., 2010; Jiang et al., 2016; Wang and Wang, 2016; Avella-Medina and Ronchetti, 2018; Salehi et al., 2019; Ma et al., 2020). ProSGPV often achieves the fastest rate of being support recovery consistent. The intuition behind its nice properties is as follows. Lasso evaluated at $\lambda_{\text{gic}}$ reduces the feature space to a candidate set that is very likely to contain the true support. This behavior is particularly important when $p > n$. The second-stage thresholding examines whether each variable is clinically meaningful enough

| | Logistic regression | | | Poisson regression | | | Cox regression | | |
|---|---|---|---|---|---|---|---|---|---|
| | Low-s | Low-d | High-s | Low-s | Low-d | High-s | Low-s | Low-d | High-s |
| n | 40:800 | 40:800 | 200 | 40:800 | 40:800 | 120 | 40:800 | 40:800 | 80 |
| p | 20 | 20 | 200:800 | 20 | 20 | 120:480 | 20 | 20 | 80:320 |
| s | 4 | 14 | 4 | 4 | 14 | 4 | 4 | 14 | 4 |
| $\beta_l$ | 0.5 | 0.5 | 0.5 | 0.1 | 0.1 | 0.1 | 0.2 | 0.2 | 0.2 |
| $\beta_u$ | 1.5 | 1.5 | 1.5 | 0.4 | 0.4 | 0.4 | 0.8 | 0.8 | 0.8 |
| t | 0 | 0 | 0 | 2 | 2 | 2 | 0 | 0 | 0 |

Table 2.1: Summary of parameters in simulation studies. Low-s stands for low dimensional sparse; low-d stands for low dimensional dense; and high-s stands for high dimensional sparse.

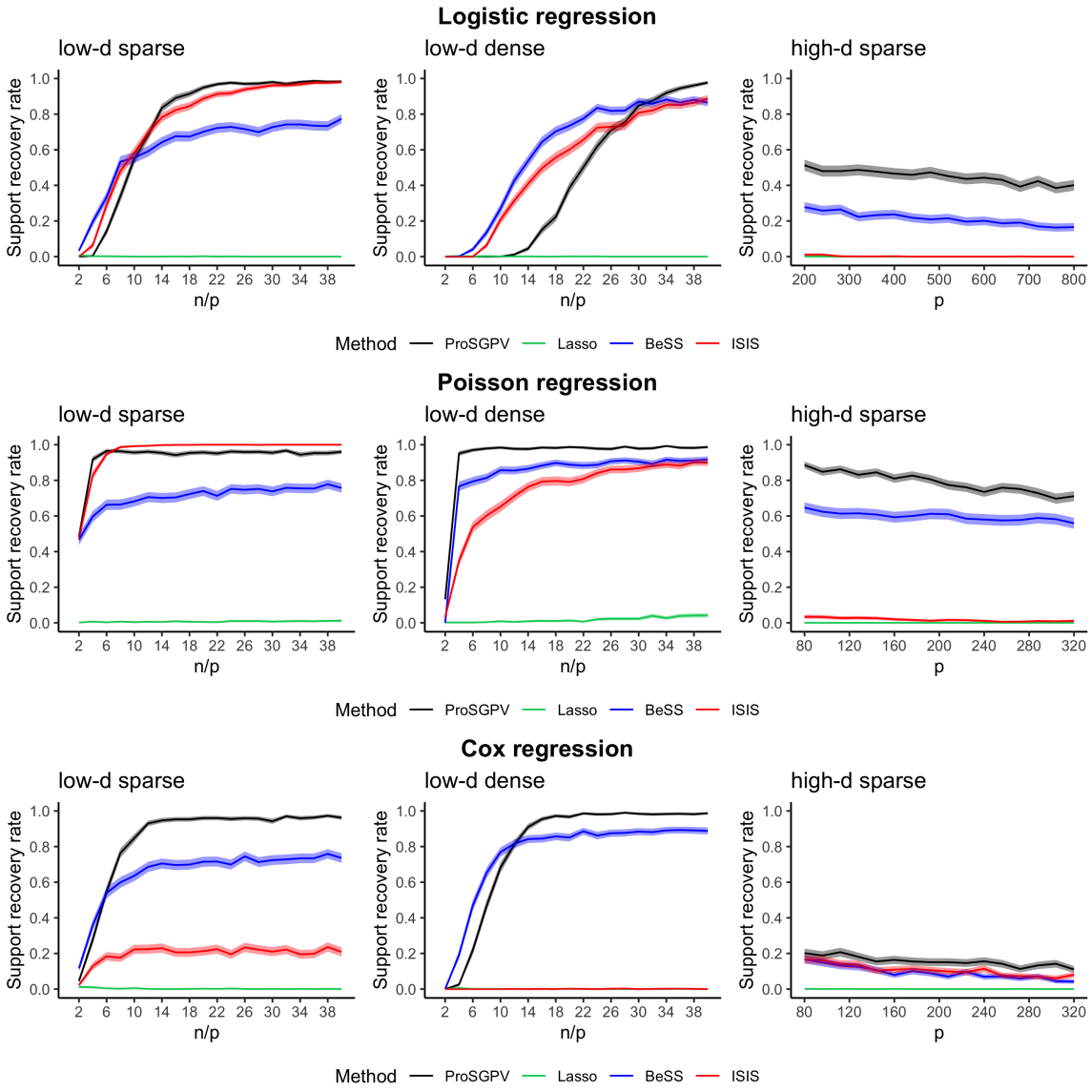


Figure 2.5: Comparison of capture rate of the exact true model: mean rates surrounded by 95% Wald confidence intervals over 1000 simulations

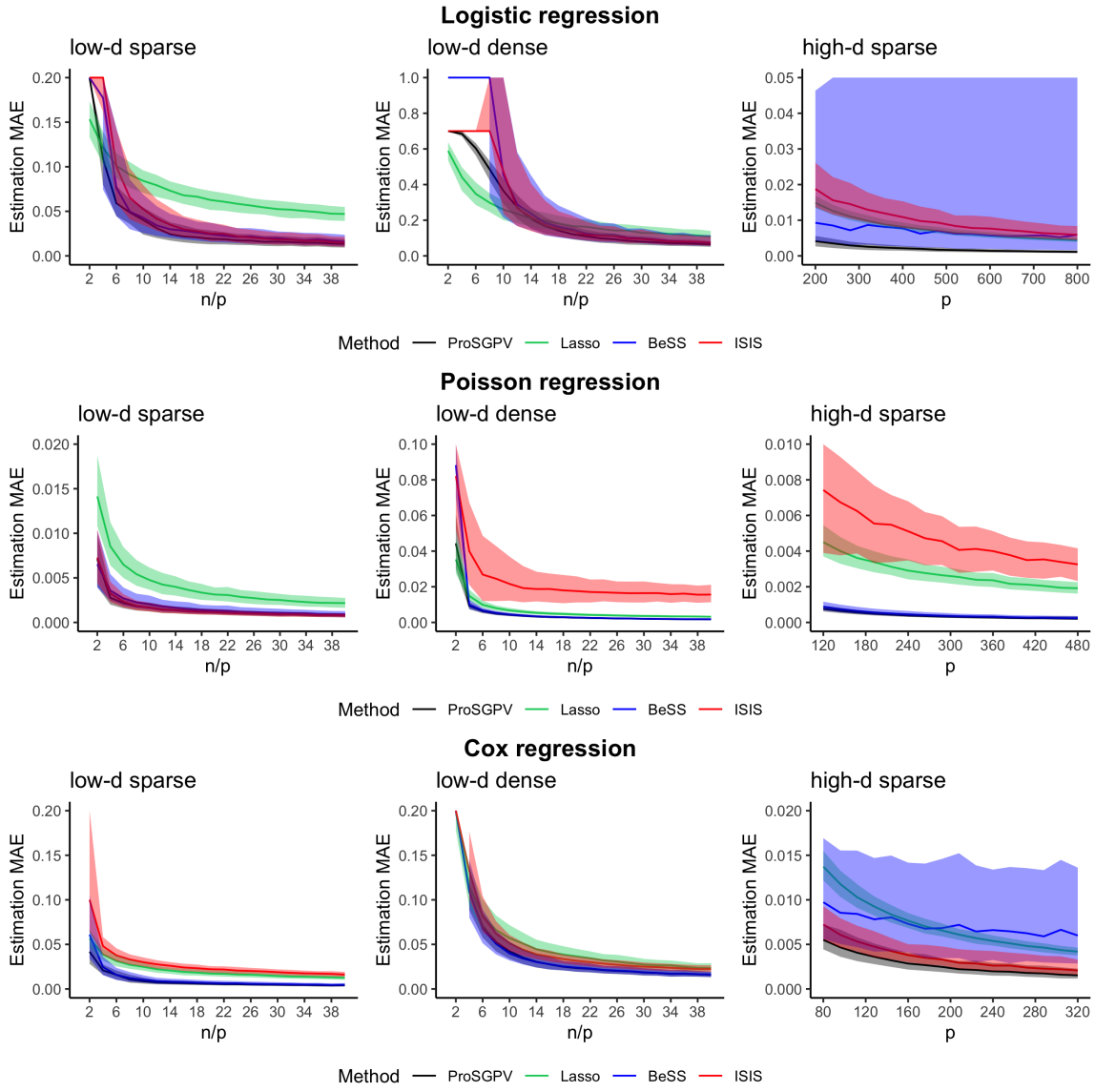


Figure 2.6: Comparison of parameter estimation: medians of mean absolute error in parameter estimation surrounded by first and third quartiles over 1000 simulations

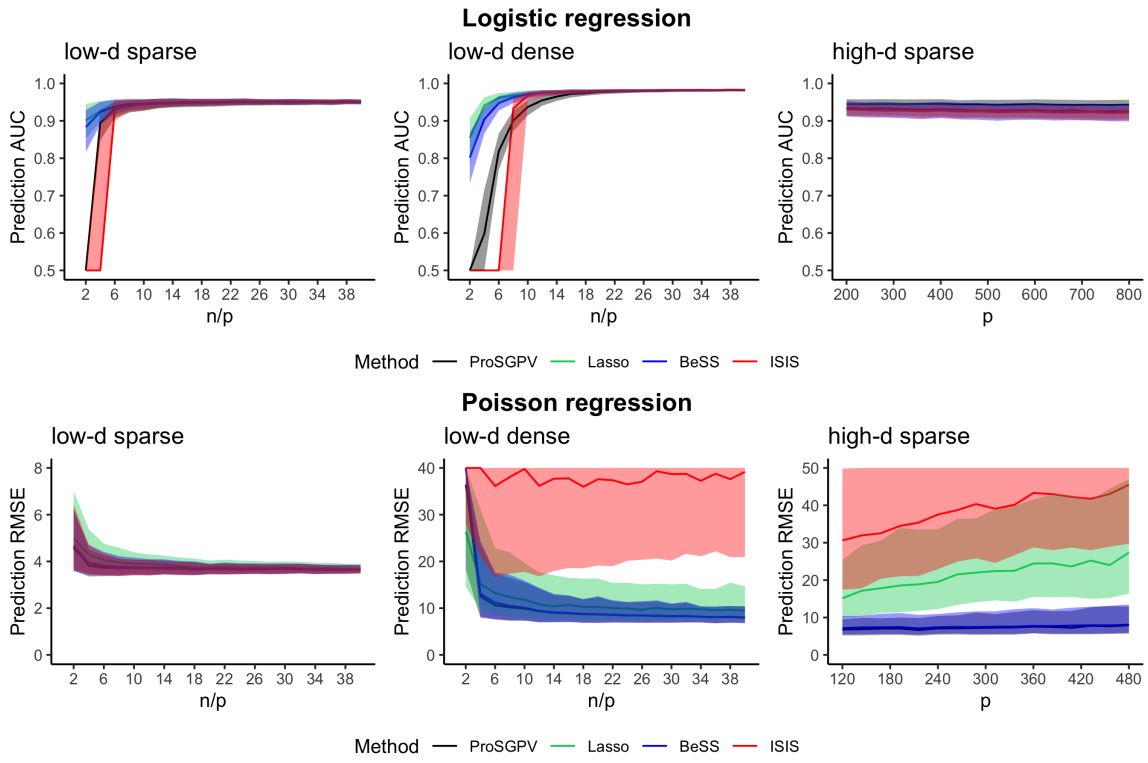


Figure 2.7: Comparison of prediction performance in a separate test set: median area under the curve surrounded by first and third quartiles in logistic regression and median prediction root mean square errors (RMSE) surrounded by first and third quartiles in Poisson regression. RMSE are bounded for aesthetic reasons.

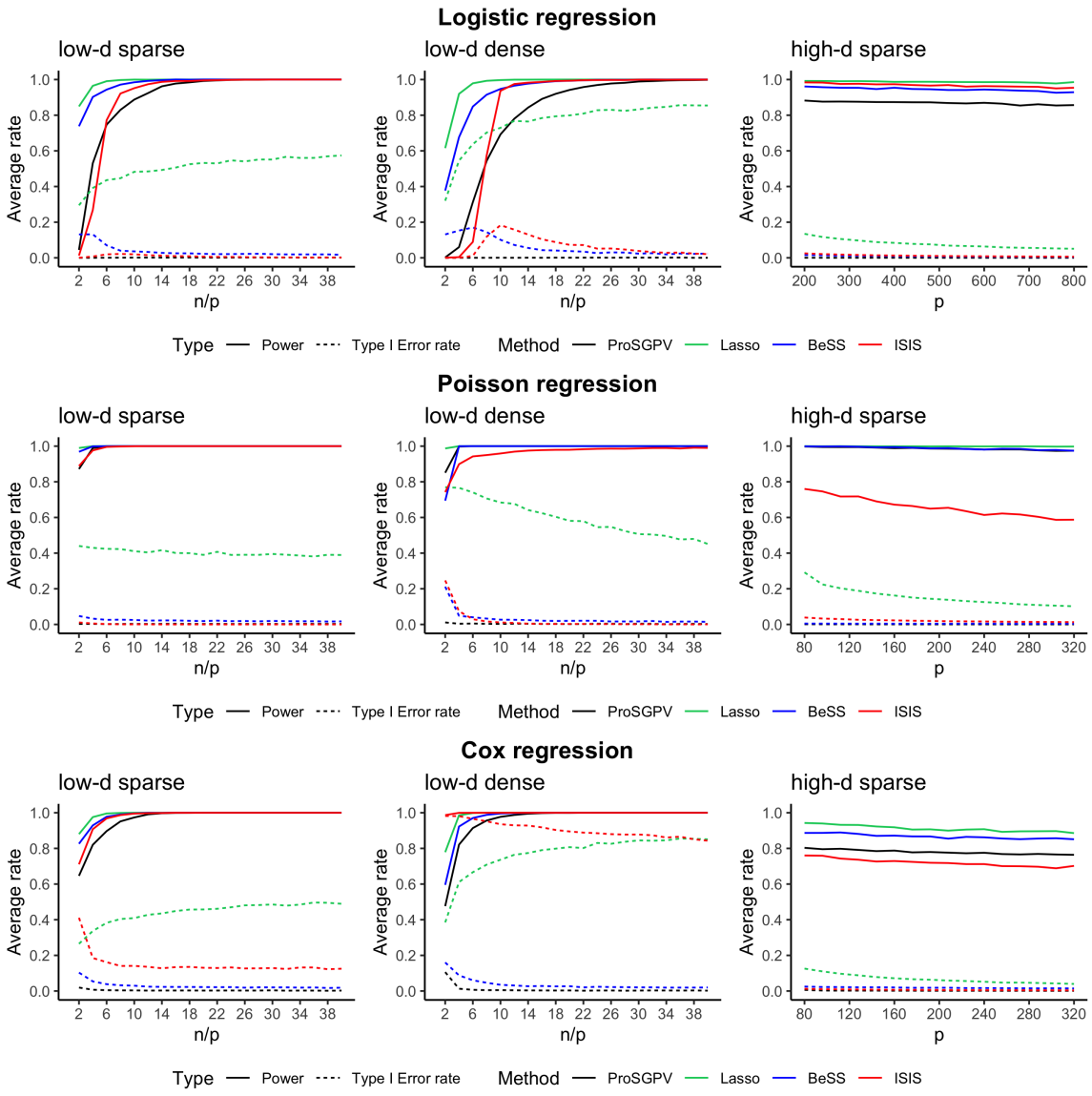


Figure 2.8: Comparison of Type I Error rate and Power for all algorithms. Average Type I Error rates and estimated power rates from 1000 simulations are compared for all algorithms. Average Type I Error rates are calculated as the average proportion of falsefully identified signal variables among all noise variables. Estimated power rates are calculated as the average proportions of correctly identified signal variables.

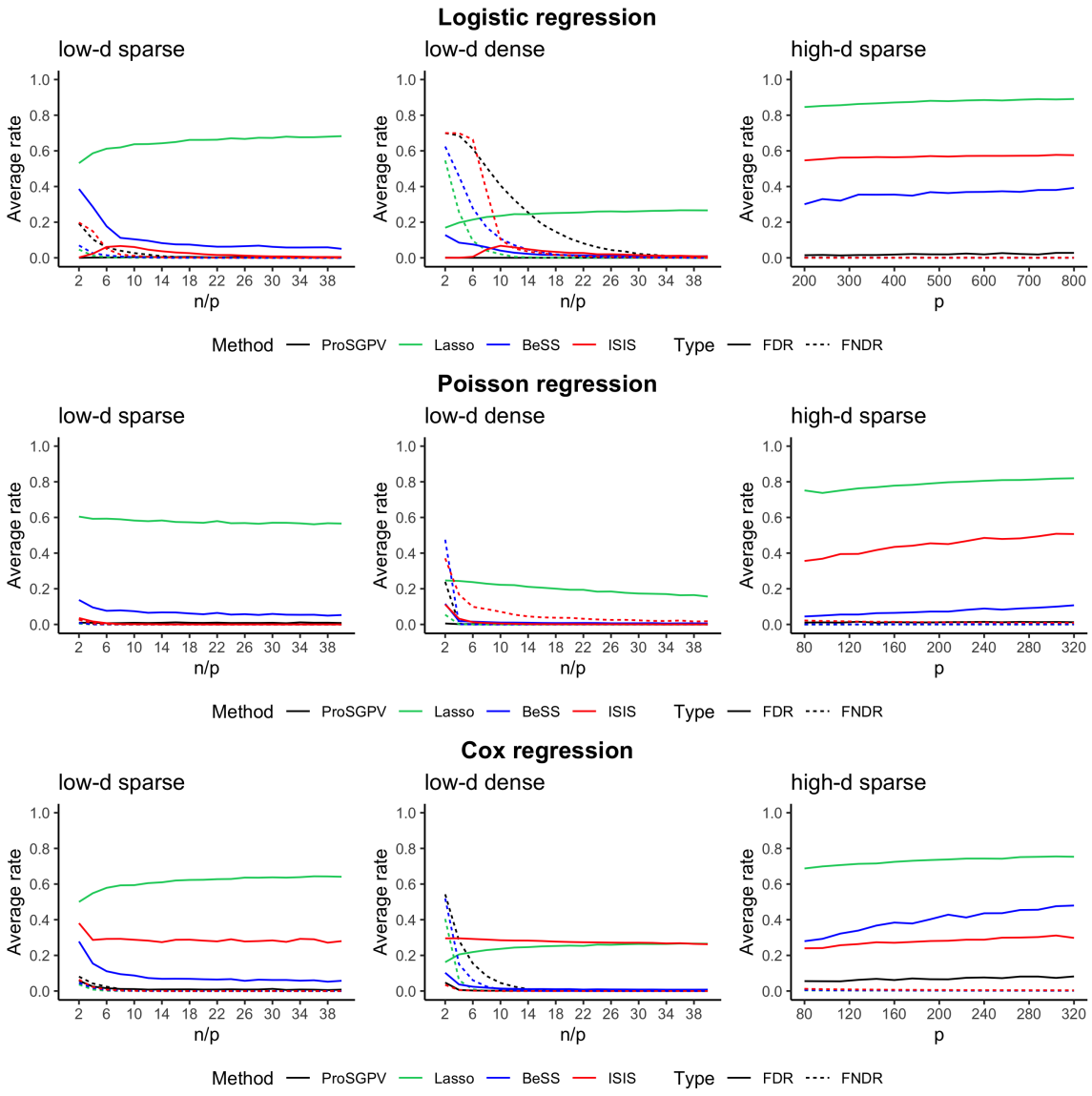


Figure 2.9: Comparison of FDR and FNDR for all algorithms. Average false discovery proportions (pFDR) and false non-discovery proportions (pFNDR) are compared for all algorithms over 1000 simulations. pFDR is calculated as the proportion of identified "signal" variables that are indeed noise variables. pFNDR is calculated as the proportion of identified "noise" variables that are indeed signal variables.

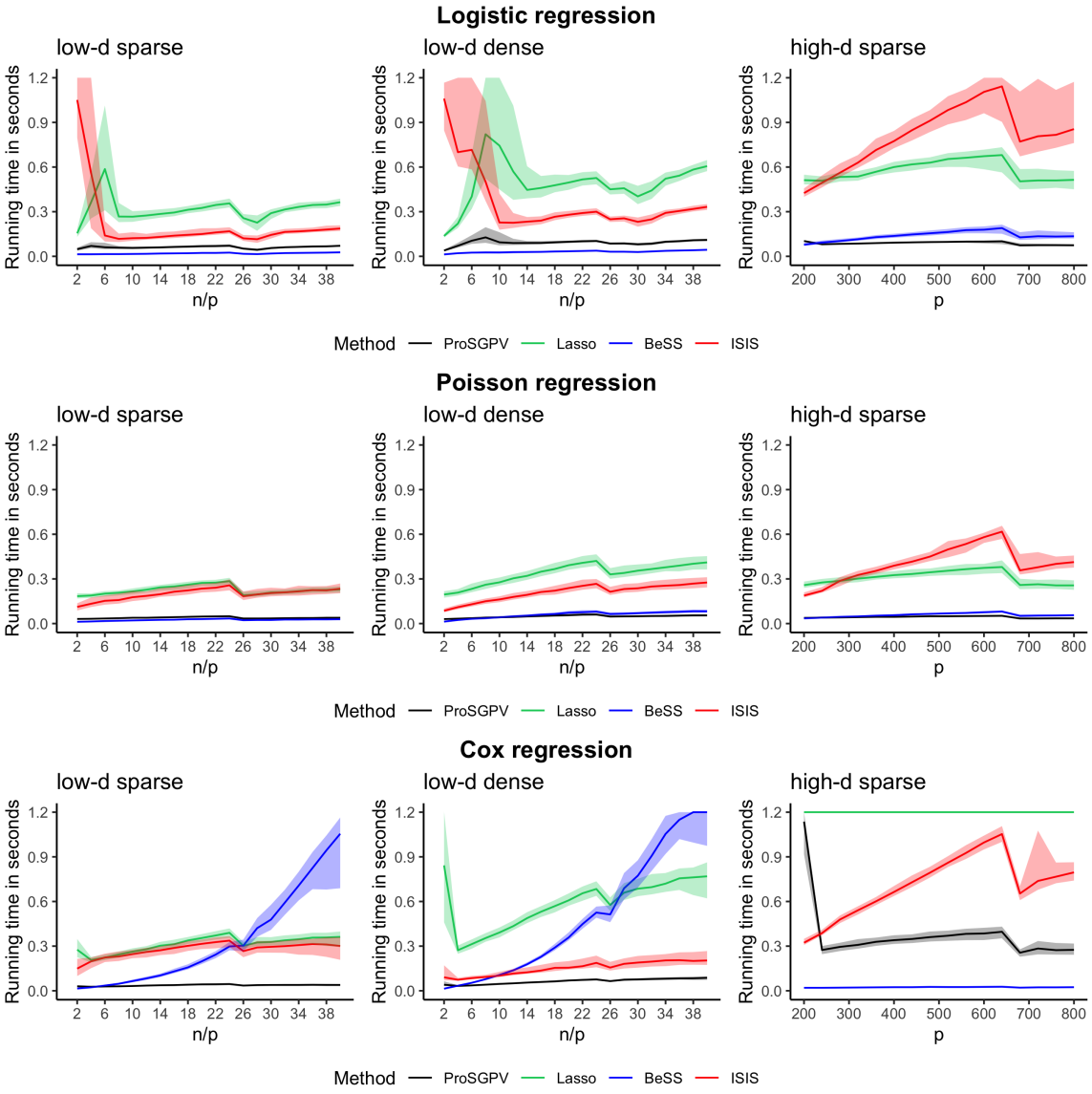


Figure 2.10: Comparison of computation time for all algorithms. Running time in seconds are compared for all algorithms over 1000 repetitions. For aesthetic reasons, data are capped at 1.2 seconds.

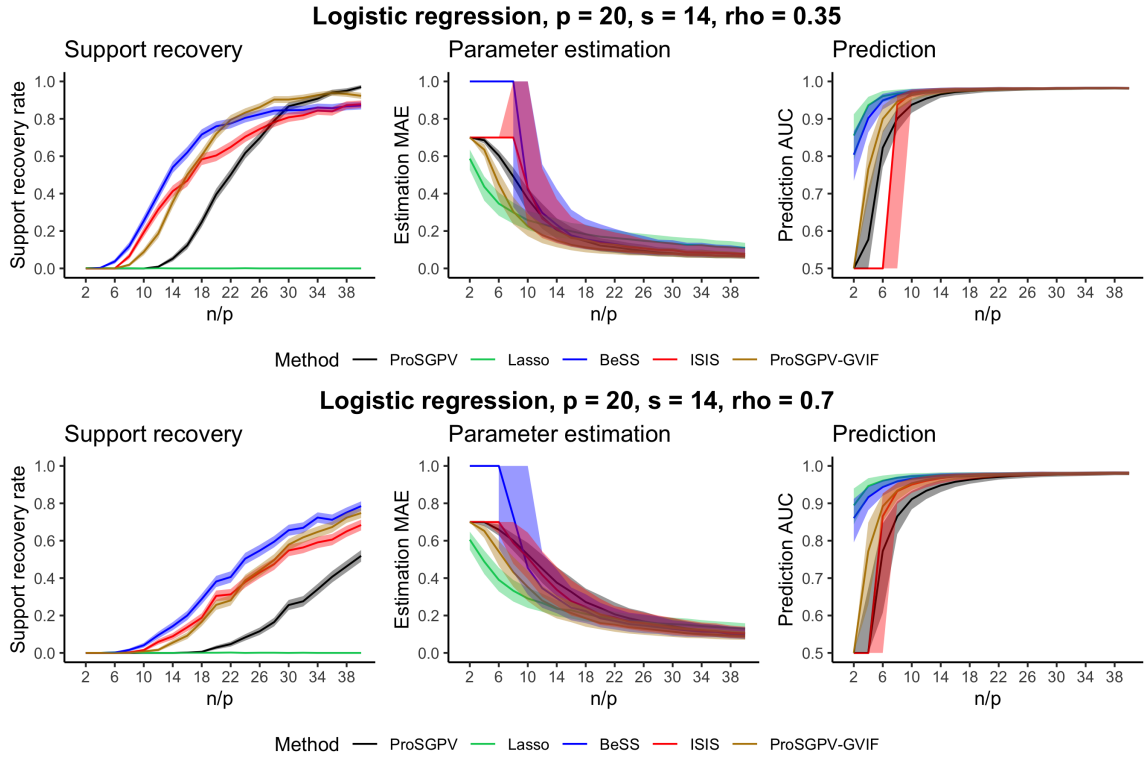


Figure 2.11: Comparison of a constant null bound and a data-dependent null bound. Support recovery rate, parameter estimation mean absolute error (MAE), and prediction area under the curve (AUC) are compared for the ProSGPV with a constant null bound and the ProSGPV with a generalized variance inflation factor adjusted null bound. Means (solid lines) and Wald 95% confidence intervals (shades) are compared for support recovery. Median (solid lines) and first and third quartiles (shades) MAEs are compared for parameter estimation. Median (solid lines) and first and third quartiles (shades) prediction AUC are compared for prediction using a separate test set.

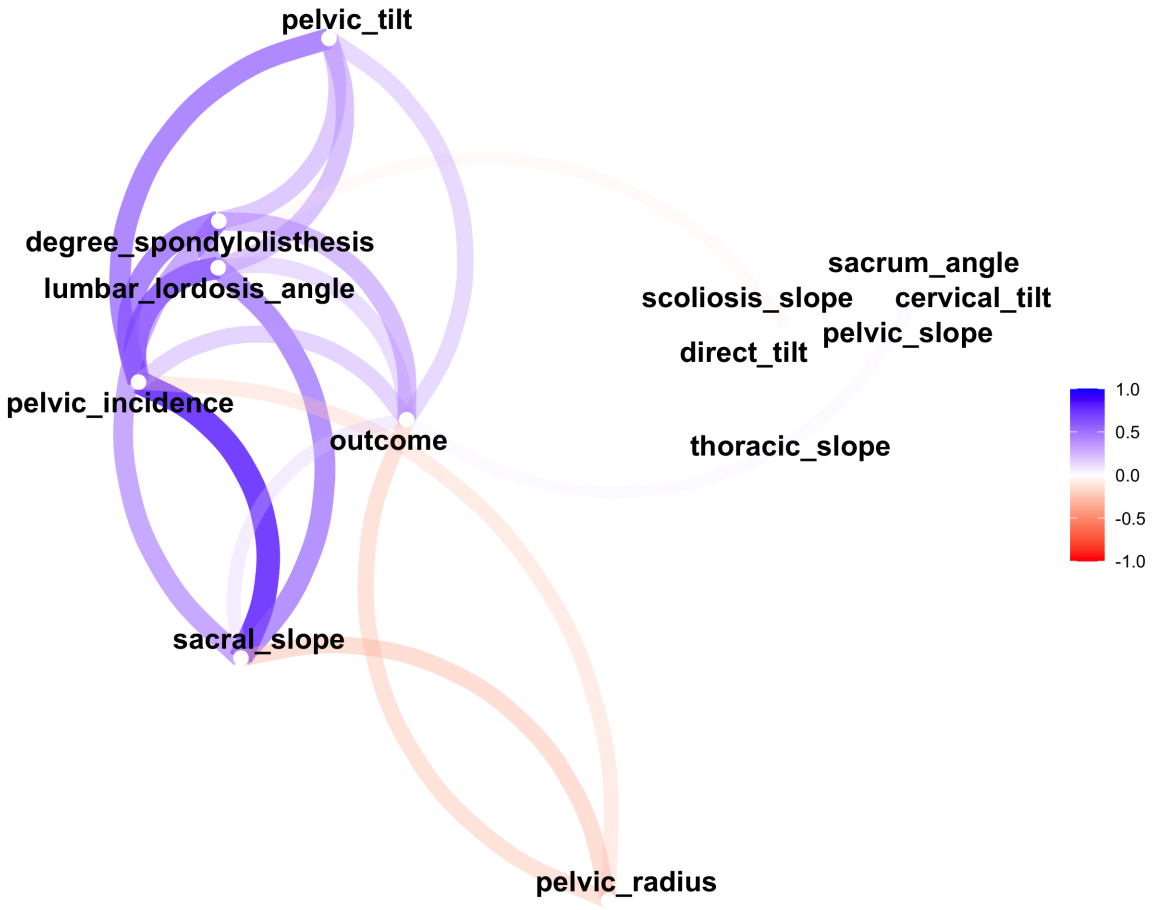


Figure 2.12: Clustering and correlation pattern of the spine data. The positions of variables imply the clustering pattern. Color indicates the strength of pairwise correlation. Blue indicates a positive correlation and red indicates a negative correlation. The darker the color, the stronger the correlation.

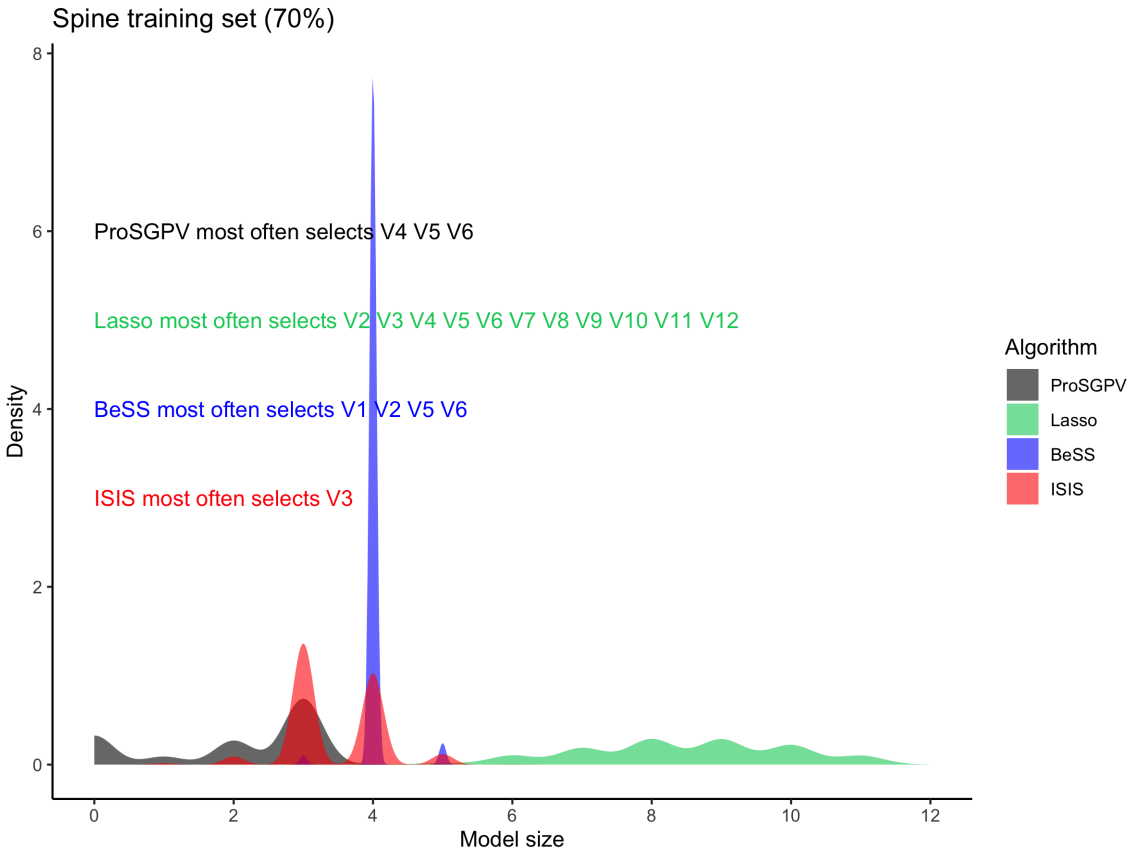


Figure 2.13: Comparison of sparsity of solutions from all algorithms. Density of model of each algorithm using the training data (70% of all data) are compared over 1000 repetitions. Most often selected models are also annotated with color.

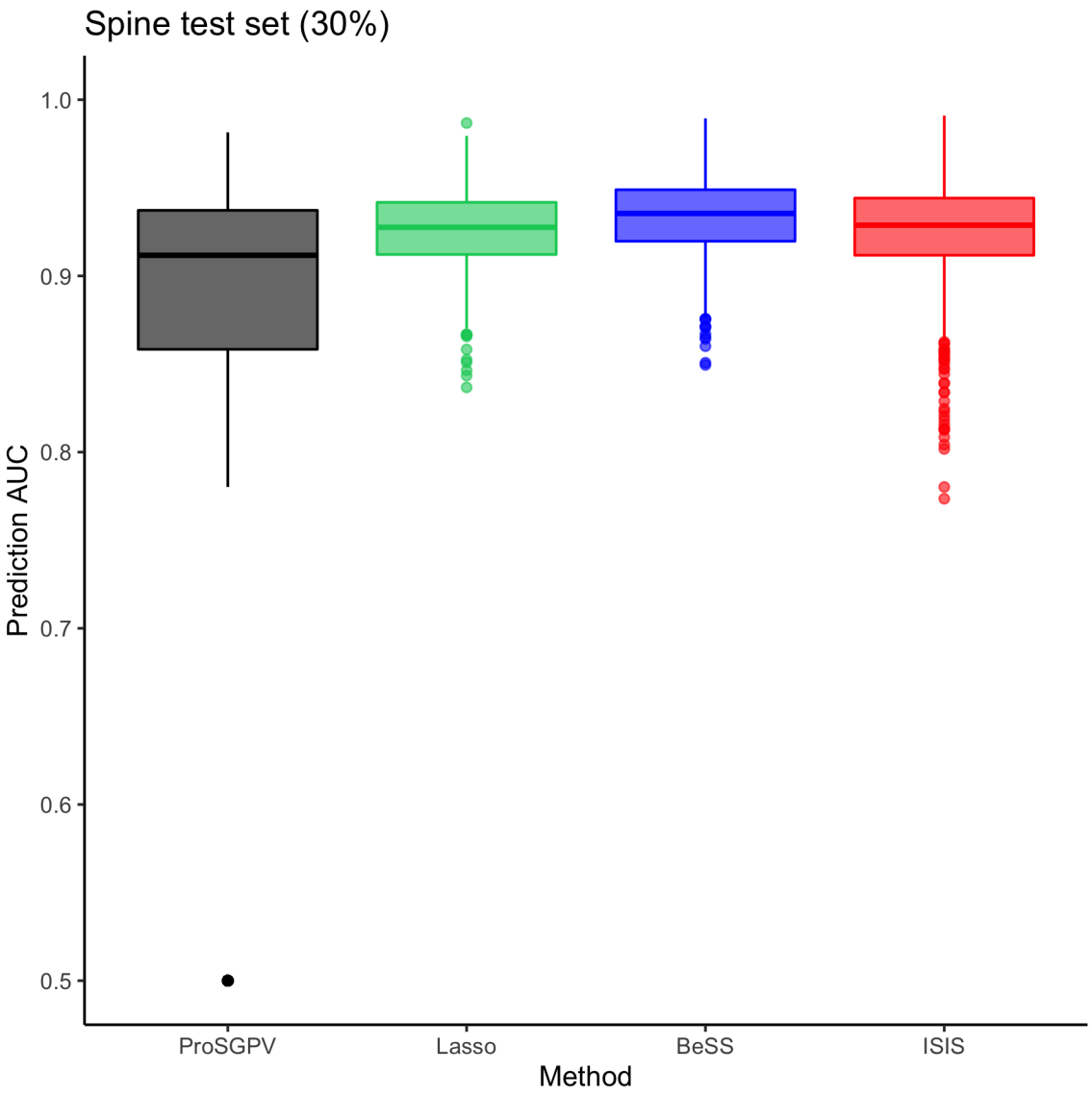


Figure 2.14: Comparison of prediction accuracy from all algorithms. Distribution of prediction area under the curve in the test set (30% of all data) of each algorithm are compared over 1000 repetitions.

to stay in the model. Some statistically significant variables may be excluded from the final model, as their effects overlap with the null zone. That explains why ProSGPV has a high power and an extremely low Type I Error rate. The successful support recovery relies on the assumption that true signals have larger effect sizes than noise variables in the data set. If that is violated, ProSGPV may miss one or two signal variables whose effects are below a clinically significant level, as measured by the null bound ($\overline{SE}$). But to be fair, no current method preforms well when that overlap exists.

ProSGPV recovers the true support with high probability and its final model does not include shrunken estimates. Therefore, the parameter estimation performance of ProSGPV tends to be excellent and superior in comparison to standard methods. We conjecture that ProSGPV enjoys oracle properties of support recovery and parameter estimation. Here, oracle properties refers to that its estimates are as good as maximum likelihood estimates when the true support is known (Fan and Li, 2001). Our simulations, combined with the real-world data example, provide strong evidence that ProSGPV, even though it is not optimized for prediction, yields comparable prediction performance to other standard procedures.

### 2.6.2 When ProSGPV fails

When signals are dense, such as in a classic logistic regression and $n$ is small, ProSGPV may select fewer signal variables than the truth, resulting in a sparse model. This could be remedied by adjusting the null bound to include more variables with smaller effect sizes, as we have shown with the GVIF adjustment. However, changing the null bound in all scenarios would needlessly affect ProSGPV's general performance. On the flip side, ProSGPV does not select any noise variables, which implies that variables selected are very likely to be true signals (low false discovery rate).

In simulation studies not shown here, ProSGPV tended to have poor support recovery performance when true effect sizes are below a certain noise level, as suggested by Section 2.6.1. It is a common assumption that the effect size of true signals are large enough to be detectable for a successful support recovery (in general, above the noise level) (Wang et al., 2010; Jiang et al., 2016; Salehi et al., 2019). In addition, we observed in Logistic regression that when the true effect size is too large all algorithms had difficulty identifying the true support. That is likely due to issues with complete/quasi-complete separation (Albert and Anderson, 1984; Rahman and Sultana, 2017). Logistic regression maps the linear predictor of explanatory variables to a continuous probability but the outcome is binary (discrete). When $n$ is small, different combinations of variables may achieve very similar probabilities for an event. Fortunately, the prediction performance does not suffer in the high dimensional dense signal setting.

### 2.6.3 Remedies when data are highly correlated or signals are dense

When the design matrix is orthogonal and variables are centered and standardized, all coefficient standard errors (SE) are the same. When data columns are correlated, SEs are inflated, resulting in larger null bound in the ProSGPV. This results in screening out more small true effects and leads to worse support recovery performance, as is shown in the logistic example in Figure 2.5. However, this can be addressed by replacing the constant null bound with a generalized variance inflation factor (GVIF) (Fox and Monette, 1992) adjusted null bound. This action deflates the SEs for variables that have high correlation with other variables, reduces the size of the null bound, and consequently includes slightly more variables in the ProSGPV selection set. The GVIF-adjusted ProSGPV exhibits improvement over the original ProSGPV when data are correlated and signals are dense (see Figure 2.11).

### 2.6.4 Closing comments

There is a rich literature in variable selection methods in Logistic regression, either in classic $n > p$ or high-dimensional $p > n$ settings, variable selection in Poisson, among other GLM, and Cox regression. Algorithms proposed for Poisson and Cox models are not easily transferrable to Logistic, or other GLM. ProSGPV appears to be a unified variable selection approach that works consistently well in linear, Logistic, Poisson, and Cox models. It reduces the feature space to a smaller candidate set and applies SGPV thresholding to select variables that are clinically meaningful. Simulation studies show that ProSGPV achieves the best support recovery and parameter estimation while not giving up much prediction performance. The ProSPGV algorithm does not depend on tuning parameters that are hard to specify. When data are highly correlated or signals are known to be dense in the data, the inference and prediction performance is improved by using a GVIF-adjusted null bound in the ProSGPV algorithm. The ProSGPV algorithm is readily adaptable (e.g., using the GVIF to modify the null bound) and is fast to compute in practice. Future endeavors could be made to characterize the rate of support recovery, parameter estimation, and prediction. The novelty of the SGPV approach is that formally incorporate estimation uncertainty into variable selection tasks. This idea readily generalizes to variable selection tasks with other data types and may lead to statistical tools for other real-world problems.

# CHAPTER 3

## Software implementation of the ProSGPV algorithm

**Abstract**

We introduce the ProSGPV R package, which implements a variable selection algorithm based on second-generation p-values (SGPV) instead of traditional p-values. Most variable selection algorithms shrink point estimates to arrive at a sparse solution. In contrast, the ProSGPV algorithm accounts for the estimation uncertainty – via confidence intervals – in the selection process. This additional information leads to better inference and prediction performance in finite sample sizes. ProSGPV maintains good performance even in the high dimensional case where $p > n$, or when explanatory variables are highly correlated. Moreover, ProSGPV is a unifying algorithm that works with continuous, binary, count, and time-to-event outcomes. No cross-validation or iterative processes are needed and thus ProSGPV is very fast to compute. Visualization tools are available in this package for assessing the variable selection process. Here we present simulation studies and a real-world example to demonstrate ProSGPV's inference and prediction performance in relation to the current standards in variable selection procedures.

## 3.1 Introduction

As the sheer volume of data grows at an astronomical rate, variable selection plays an increasingly crucial role in research. This is particularly true in the high dimensional setting where $p > n$ and classical statistical methods exploiting the full feature space no longer work. An ideal variable selection procedure would recover the underlying true support with high probability, yield parameter estimation with low bias, and achieve good prediction performance. While it is hard for a statistical procedure to strike a balance between inference and prediction tasks (Meinshausen et al., 2006; Shmueli et al., 2010), the ProSGPV algorithm is remarkably able in this sense (Zuo et al., 2021b,a).

One natural approach to variable selection is best subset selection (BSS). BSS chooses $k$ out of $p$ total variables for each $k \in \{1, 2, ..., p\}$ that maximize a chosen loss function. It can be thought of as an $\ell_0$-penalized regression. Natarajan, 1995 showed that the BSS problem is nonconvex and NP-hard. With recent advancements (Hazimeh and Mazumder, 2020; Di Gangi et al., 2019; Weng et al., 2019), solving the BSS routine with thousands of features is no longer infeasible. Particularly, an efficient R package called **BeSS** (Weng et al., 2019) is scalable to identify the best sub-model in seconds or a few minutes when $p$ is around 10,000. $\ell_1$-penalized likelihood procedures are also used for variable selection. Lasso, for example, produces models with a strong predictive ability (Tibshirani, 1996). However, lasso is not always variable selection consistent (Leng et al., 2006; Meinshausen et al., 2006). To address the inconsistency issue, adaptive lasso was proposed, which introduces weights in the $\ell_1$ penalty (Zou, 2006). Despite oracle variable selection properties of the adaptive lasso, it is often hard in practice to find a tuple of tuning parameters that achieve the properties. Both lasso and adaptive lasso can be implemented in the **glmnet** package (Friedman et al., 2010; Simon et al., 2011). Smoothly clipped absolute deviation (SCAD) (Fan and Li, 2001) and minimax concave penalty with penalized linear unbiased selection (MC+) (Zhang et al., 2010) were proposed to bridge the gap between the $\ell_0$ and $\ell_1$ penalties. The two algorithms are largely distinguished by their piecewise linear thresholding functions. Sure independence screening (SIS) (Fan and Lv, 2008; Fan and Song, 2010), implemented in the **SIS** package (Saldana and Feng, 2018), ranks the maximum marginal likelihood estimates and can greatly reduce the dimensionality of feature space by keeping top variables in the ranking, even when $p \gg n$. Iterative SIS (ISIS) can improve its performance in finite sample sizes. Note that all aforementioned algorithms shrink point estimates to derive a sparse solution and there is room for improvement of inference and prediction properties in finite sample sizes.

Many R packages have been developed to address certain data types. For example, **clogitL1** (Reid and Tibshirani, 2014) performs variable selection with lasso and elastic net penalties in conditional logistic regression; **pogit** (Dvorzak and Wagner, 2016) performs Bayesian variable selection with spike and slab priors

in Poisson and Logistic regressions; **penPHcure** (Beretta and Heuchenne, 2021) performs variable selection in Cox proportional hazards cure model with time-varying covariates. The ideal R package would have superior or comparable performance as the current algorithms, and work with each of continuous, binary, count, and time-to-event outcomes.

Recently, Zuo et al., 2021b,a developed Penalized Regression with Second Generation P-Values (ProSGPV) for variable selection in both low-dimensional ($n > p$) and high-dimensional ($p > n$) settings. Unlike traditional algorithms, ProSGPV incorporates estimation uncertainty via confidence intervals in the variable selection process. This addition often leads to better support recovery, parameter estimation, and prediction performance in linear regression, GLM, and Cox regression. This paper describes an R package named **ProSGPV**, which implements the ProSGPV algorithm. It is a user-friendly tool to perform variable selection and visualize the process and results. Simulation studies below compare the inference and prediction performance of **ProSGPV** against that of **glmnet**, **BeSS**, and **ISIS**, in scenarios not discussed in Zuo et al., 2021b,a. A real-world example compares the sparsity of solutions and prediction accuracy of all algorithms.

### 3.2 Methods

#### 3.2.1 What is a second-generation p-value?

Second-generation p-values (SGPVs), denoted as $p_\delta$, attempt to resolve some of deficiency of traditional p-values by replacing the point null with a pre-specified interval null (Blume et al., 2018, 2019). The interval null is the set of effects that are scientifically indistinguishable or immeasurable from the point null hypothesis, due to limited precision or practicality. SGPVs are defined as the fraction of data-supported hypotheses that are also null hypotheses. An illustration of how SGPVs work is shown in Figure 3.1.

Formally, let $\theta$ be a parameter of interest, and let $I = [\theta_l, \theta_u]$ be an interval estimate of $\theta$ whose length is given by $|I| = \theta_u - \theta_l$. In this paper we will use a 95% CI for $I$, but any uncertainty interval can be used. If we denote the length of the interval null by $|H_0|$, then the SGPV $p_delta$ is defined as

$$p_\delta = \frac{|I \cap H_0|}{|I|} \times \max\left\{\frac{|I|}{2|H_0|}, 1\right\} \tag{3.1}$$

where $I \cap H_0$ is the intersection of two intervals. The correction term $\max\{|I|/(2|H_0|), 1\}$ fixes the problem when the interval estimate is very wide, i.e., when $|I| > 2|H_0|$. In that case, the data are effectively inconclusive and the correction term shrinks the SGPV back to 1/2. As such, SGPV indicate when data are compatible with null hypotheses ($p_\delta = 1$), or with alternative hypotheses ($p_\delta = 0$), or when data are inconclusive ($0 < p_\delta < 1$). By design, SGPVs emphasize effects that are clinically meaningful by exceeding a pre-specified null level. Empirical studies have shown SGPVs have potential for identifying feature impor-

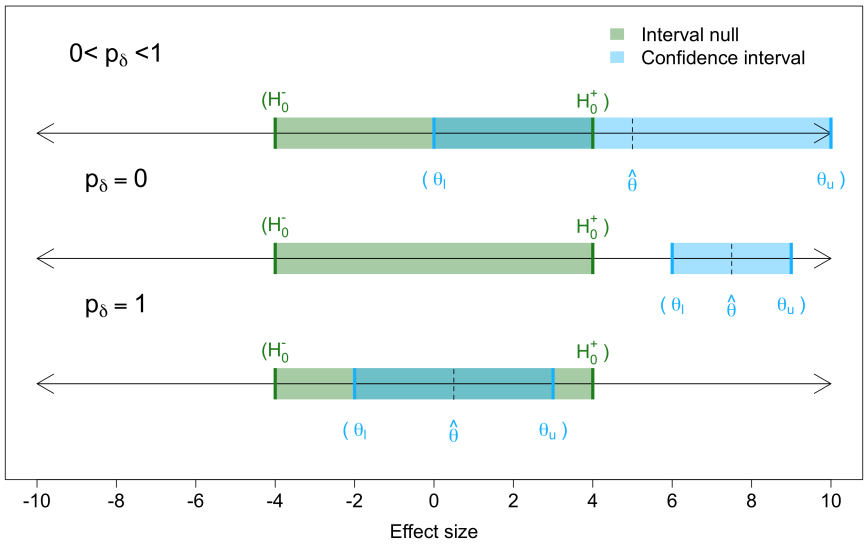


Figure 3.1: An illustration of how SGPVs work

tance in high dimensional settings (Blume et al., 2018, 2019; Zuo et al., 2021b,a). The null bound in the SGPVs is typically the smallest effect that would be clinically relevant or the effect magnitude that can be distinguished from noise, on average. Zuo et al., 2021b,a proposed using a generic null interval for regression coefficients that shrinks to zero and is based on the observed level of noise in the data. This extends the null bound in Blume et al., 2018, 2019 that remains constant. The interval is easily obtained in the variable selection step and promotes good statistical properties in the selection algorithm.

### 3.2.2 The ProSGPV algorithm

The ProSGPV algorithm is a two-stage approach where in the first stage a candidate set of variables is acquired, and in the second stage further thresholding is applied to select variables that are ”truly” associated with the outcome.

The steps of the general ProSGPV algorithm are shown below in Algorithm 3.

**Algorithm 3** The general ProSGPV algorithm

**procedure** PROSGPV($X$, $Y$)
    **Stage one**: Find a candidate set
        Fit a lasso and evaluate it at $\lambda_{\text{gic}}$
        Fit OLS/GLM/Cox models on the lasso active set
    **Stage two**: SGPV screening
        Extract the confidence intervals of all variables from the previous step
        Calculate the mean coefficient standard error $\overline{SE}$
        Calculate the SGPV for each variable where $I_j = \hat{\beta}_j \pm 1.96 \times SE_j$ and $H_0 = [-\overline{SE}, \overline{SE}]$
        Keep variables with SGPV of zero
        Refit the OLS/GLM/Cox with selected variables
**end procedure**

By default, data are scaled and centered in linear regression but are not transformed as such in GLM and Cox regression. No notable difference is observed when data are standardized in GLM and Cox regression. In the first stage, lasso is used to reduce the feature space to a candidate set that is very likely to contain true signals. This pre-screening is crucial to high dimensional data ($n < p$) and improves the support recovery and parameter estimation in low dimensional data (Zuo et al., 2021b). The lasso is evaluated at $\lambda_{\text{gic}}$, but the algorithm is robust with respect to the choice of $\lambda$ (Zuo et al., 2021b). In the second stage, the null bound is set to be the mean standard error of all coefficient estimates. However, the algorithm is insensitive to any reasonable scale change in the null bound (Zuo et al., 2021b,a). When data are highly correlated, a generalized variance inflation factor (GVIF) (Fox and Monette, 1992) adjusted null bound can be used to improve the inference and prediction performance (Zuo et al., 2021a).

In essence, ProSGPV is a hard thresholding algorithm that shrinks small effect to zero and reserve the large effect to obtain an unbiased estimate when the true support is successfully recovered. Notation-wise, the solution to the ProSGPV algorithm $\hat{\beta}^{prosgpv}$ is

$$\begin{gathered} \hat{\beta}^{\text{prosgpv}} = \hat{\beta}^{m}_{|S} \in \mathbb{R}^p, \text{ where} \\ S = \{k \in C : |\hat{\beta}^{m}_{k}| > \lambda_k\}, C = \{j \in \{1,2,...,p\} : |\hat{\beta}^{\text{lasso}}_{j}| > 0\} \end{gathered} \tag{3.2}$$

where $\hat{\beta}^{m}_{|S}$ is a vector of length $p$ with non-zero elements being the OLS/GLM/Cox coefficient estimates from the model with variables only in the set $S$. $S$ is the final selection set and $C$ is the candidate set from the first-stage screening. Note that the cutoff $\lambda_k = 1.96 * SE_k + \overline{SE}$.

When $\lambda_{\text{gic}}$ in Algorithm 2 is replaced with zero in the first stage, ProSGPV reduces to a one-stage algorithm. That amounts to calculating SGPVs for each variable in the full model and select variables whose effects are above the threshold. However, the support recovery and parameter estimation performance of the one-stage algorithm is slightly worse than that of the two-stage algorithm (Zuo et al., 2021b). Moreover, the one-stage algorithm is not applicable when $p > n$, i.e. when the full OLS/GLM/Cox model is not identifiable.

The ProSGPV package is publicly available from the Comprehensive R Archive Network (CRAN) and a development version is available on its GitHub page https://github.com/zuoyi93/ProSGPV. The main function `pro.sgpv` implements the default two-stage and optional one-stage ProSGPV algorithm. User-friendly `print`, `coef`, `summary`, `predict`, and `plot` functions are equipped with `pro.sgpv` for both one- and two-stage algorithms. Jeffreys prior penalized logistic regression (Kosmidis and Firth, 2021) is used when the outcome is binary to stabilize coefficient estimates in the case of complete/quasi-complete separation. In the next section, we demonstrate how `pro.sgpv` works with simulated continuous outcome data.

### 3.2.3 Operation

We first present an example by applying the ProSGPV algorithm to a simulated dataset by use of `gen.sim.data` function. With sample size $n = 100$, number of variables $p = 20$, number of true signals $s = 4$, smallest effect size $\beta_{\text{min}} = 1$, largest effect size $\beta_{\text{max}} = 5$, autoregressive correlation $\rho = 0.2$ and variance $\sigma^2 = 1$, signal-to-noise ratio (SNR), defined in Hastie et al., 2020, $\nu = 2$, we generate outcomes $Y$ following Gaussian distribution. `gen.sim.data` outputs $X$, $Y$, indices of true signals, and a vector of true coefficients.

```
> library(ProSGPV)
> set.seed(1)
> sim.data <- gen.sim.data(n = 100, p = 20, s = 4, family = "gaussian",
                           beta.min = 1, beta.max = 5, rho = 0.2, nu = 2)
> x <- sim.data[[1]]
> y <- sim.data[[2]]
> (true.index <- sim.data[[3]])

[1] 2 4 5 7

> true.beta <- sim.data[[4]]
```

By default, the two-stage algorithm is used in ProSGPV. `pro.sgpv` function takes inputs of explanatory

variables `x`, outcome `y`, outcome type `family` (default is "gaussian"), `stage` indicator (default is 2), and a `GVIF` indicator (default is FALSE). A `print` method is available to show labels of the variables selected by ProSGPV.

```
> sgpv.out.2 <- pro.sgpv(x,y)
> sgpv.out.2

Selected variables are V2 V4 V5 V7
```

The variable selection process can be visualized by using the plot function. Figure 3.2 shows the fully relaxed lasso path along a range of $\lambda$s. The shaded area is the null zone and any effect whose 95% confidence interval overlaps with the null region will be considered irrelevant or insignificant. ProSGPV is evaluated at $\lambda_{\text{gic}}$. The `lpv` argument can be used to choose to display one line per variable (the confidence bound that is closer to the null region) or three lines per variable (an point estimate and confidence bounds, default). The `lambda.max` argument control the limit of the x-axis in the plot.

`summary` function outputs the regression summary of the selected model. When the outcome is continuous, an OLS is used.

```
> summary(sgpv.out.2)

Call:
lm(formula = Response ~ ., data = data.d)

Residuals:
    Min      1Q  Median      3Q     Max
-9.8066 -3.5912  0.0817  3.1530  9.5042

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.02359    0.48792  -0.048  0.96154
V2          -4.19922    0.46775  -8.978 2.53e-14 ***
V4           1.59168    0.52876   3.010  0.00334 **
V5           2.66004    0.47441   5.607 2.01e-07 ***
```

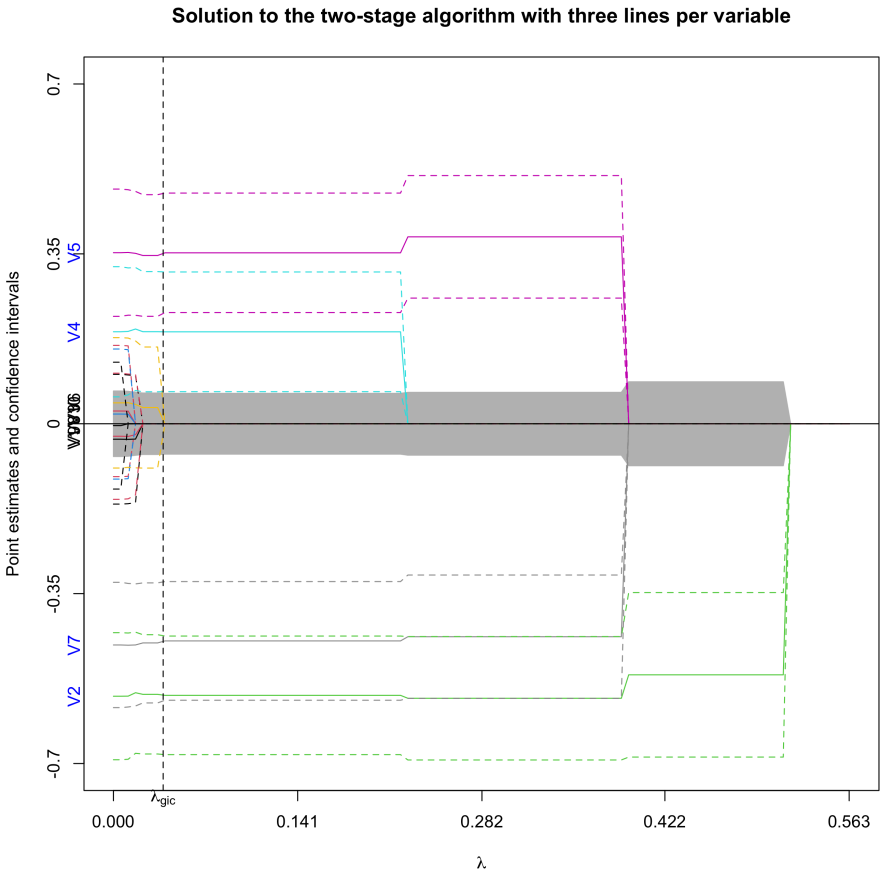


Figure 3.2: Solution path of the two-stage ProSGPV algorithm

```
V7          -3.31497    0.46207  -7.174 1.58e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.834 on 95 degrees of freedom
Multiple R-squared:  0.6367,Adjusted R-squared:  0.6214
F-statistic: 41.63 on 4 and 95 DF,  p-value: < 2.2e-16
```

`coef` function can be used to extract the coefficient estimates of length $p$. When signals are sparse, some of estimates are zero. A comparison shows that the estimates are close to the true effect sizes in the simulation.

```
> beta.hat <- coef(sgpv.out.2)
> rbind(beta.hat, true.beta)
```

```
           [,1]      [,2] [,3]     [,4]     [,5] [,6]      [,7] [,8] [,9] [,10]
beta.hat      0 -4.199224    0 1.591675 2.660038    0 -3.314971    0    0     0
true.beta     0 -5.000000    0 1.000000 2.333333    0 -3.666667    0    0     0
```

`predict` function can be used to predict outcomes using the selected model. In-sample prediction can be made by calling `predict(sgpv.out.2)` and out-of-sample prediction can be made by feeding new data set into the `newdata` argument.

In addition to the two-stage algorithm, one-stage algorithm can also be used to select variables when n¿p. The computation time is shorter for the one-stage algorithm at the expense of slightly reduced support recovery rate in the limit, as shown in Zuo et al., 2021b. Figure 3.3 shows the variable selection result of the one-stage algorithm on the same data. The one-stage algorithm missed V4 and only selected three variables. The lower confidence bound of the estimate for V4 barely excludes the null region, and was dropped from the final model because of that. The one-stage algorithm illustrates how estimation uncertainty via confidence intervals (horizontal segments) can be incorporated in variable selection.

```
> sgpv.out.1 <- pro.sgpv(x,y,stage=1)
```

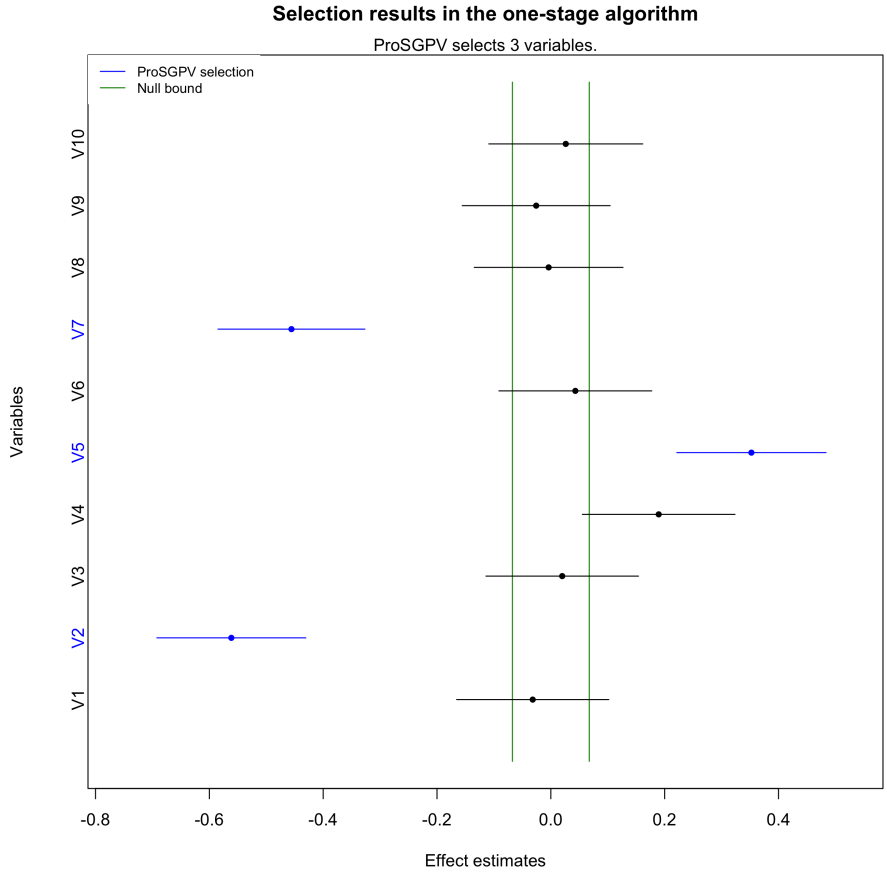


Figure 3.3: Visualization of variable selection results of the one-stage ProSGPV algorithm

```
> sgpv.out.1
Selected variables are V2 V5 V7

> plot(sgpv.out.1)
```

Examples of binary, count, and time-to-event data can be found in the package vignettes.

### 3.3 Simulation

Simulation design is adapted from Zuo et al., 2021b,a and we will present below scenarios not discussed in those two papers. The setup can be found in the Table 3.1 below. The scale and shape parameters are used to generate time-to-event from a Weibull distribution. ProSGPV is compared against lasso, BeSS, and ISIS. Results are aggregated over 1000 simulations. Evaluation metrics include support recovery rate, parameter estimation mean absolute error (MAE), prediction root mean square error (RMSE) and area under the curve in

Table 3.1: Summary of parameters in simulation studies.

| | Linear regression | Logistic regression | Poisson regression | Cox regression |
|---|---|---|---|---|
| $n$ | 100 | 32:320 | 40 | 80:800 |
| $p$ | 100:1000 | 16 | 40:400 | 40 |
| $s$ | 10 | 6 | 4 | 20 |
| $\beta_l$ | 1 | 0.4 | 0.2 | 0.3 |
| $\beta_u$ | 2 | 1.2 | 0.5 | 1 |
| $\rho$ | 0.3 | 0.6 | 0.3 | 0.3 |
| $\sigma$ | 2 | 2 | 2 | 2 |
| $\nu$ | 2 | | | |
| Intercept $t$ | | 0 | 2 | 0 |
| Scale | | | | 2 |
| Shape | | | | 1 |
| Rate of censoring | | | | 0.2 |

a separate test set. Support recovery is defined as capturing the exact true support, not containing. An estimate of MAE is $1/p\sum_{j=1}^{p}\|\hat{\beta}_j - \beta_{0,j}\|$, where $\beta_{0,j}$ is the $j$th true coefficient. Simulation results are summarized in Figure 3.4.

From Figure 3.4, we observe similar results as in Zuo et al., 2021b,a. ProSGPV often has the highest capture rates of the exact true model, has lowest parameter estimation bias, has one of the lowest prediction error, and is the fastest to compute. Note that GVIF-adjusted null bound is used in the Logistic regression because of the high correlation in the design matrix.

### 3.4 Real-world data example

In this section, we compare the performance of ProSGPV with lasso, BeSS, and ISIS algorithms using a real-world financial data. The close price of Dow Jones Industrial Average (DJIA) was documented from Jan 1, 2010 to November 15, 2017 and eight groups of primitive, technical indicators, big U.S. companies, commodities, exchange rate of currencies, future contracts and worlds stock indices, and other sources of information (Hoseinzade and Haratizadeh, 2019) were collected to predict the DJIA close price. In the analyzed data with complete records, there are 1114 observations and 82 predictors. We randomly sampled 614 observations as a fixed test set to evaluate the prediction performance of models built on the training set. We allowed the training set size n to vary from 40 to 300. At each $n$, we recorded the distribution of the training model size for each algorithm as well as the distribution of the prediction RMSE over 1000 repetitions. Results are summarized in Figure 3.5. ProSGPV and lasso had sparser models than BeSS and ISIS did, while ProSGPV had much better prediction performance in the test set. BeSS and ISIS had better prediction performance at the cost of including much more variables in the final model. The tradeoff between the sparsity of solutions and prediction accuracy is well illustrated in this example. Variables frequently

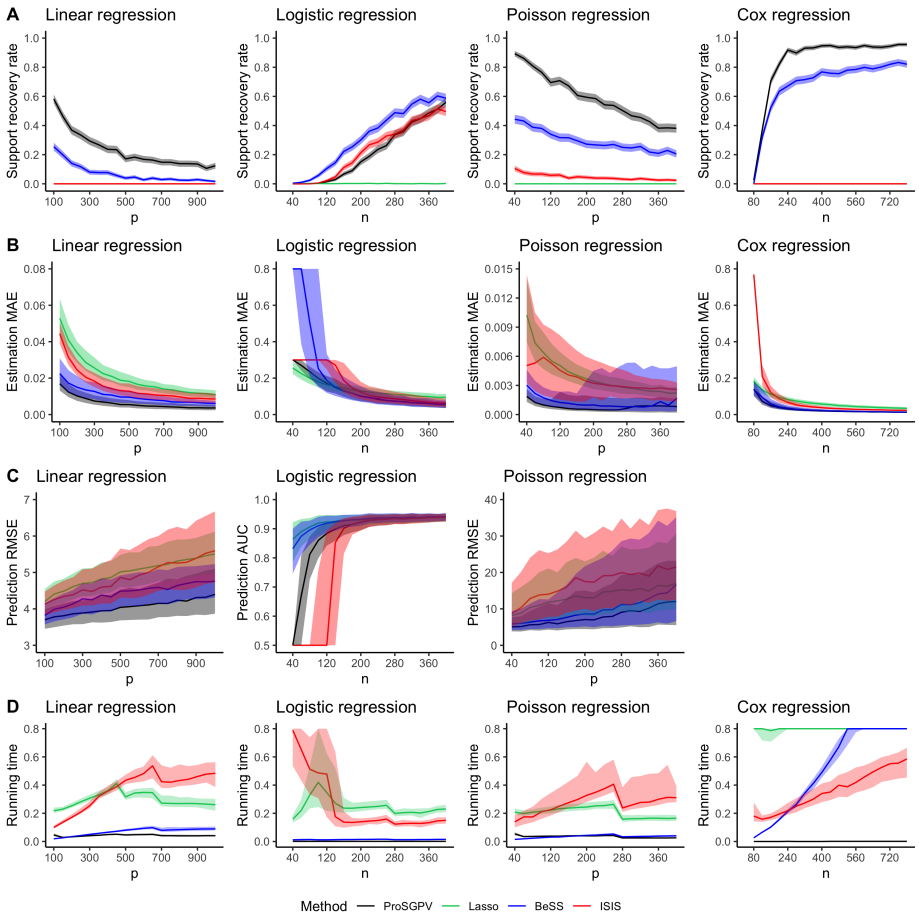


Figure 3.4: Simulation results over 1000 iterations. A) Average support recovery rate, means surrounded by 95% Wald confidence intervals. B) Average mean absolute error, medians surrounded by 1$^{st}$ and 3$^{rd}$ quartiles. C) Prediction accuracy in a separate test set (root mean square error for linear and Poisson regressions, and area under the curve for Logistic regression), medians surrounded by 1$^{st}$ and 3$^{rd}$ quartiles. D) Average running time in seconds, medians surrounded by 1st and 3rd quartiles. Values are censored at 0.8 seconds for aesthetic reasons.

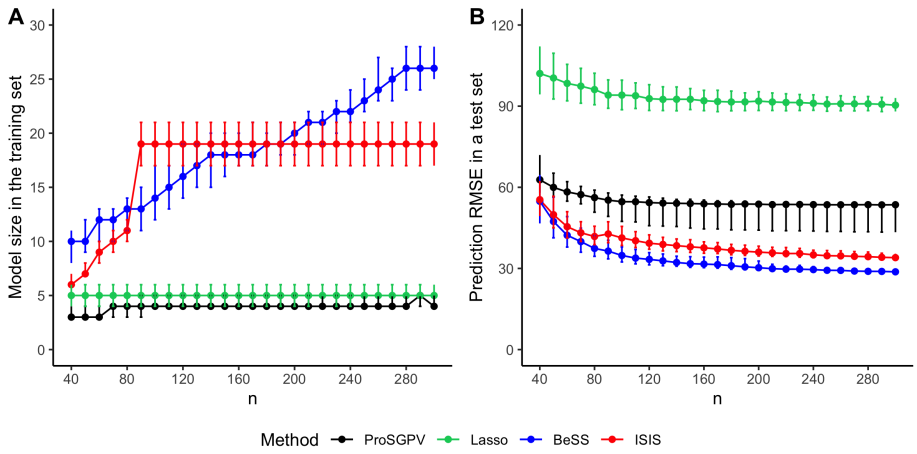


Figure 3.5: Sparsity of solutions (A) and prediction performance (B) of all algorithms in the real-world example. Medians as well as 1st and 3rd quartiles from 1000 repetitions are compared.

selected by ProSGPV include 5-, 10-, and 15-day rate of change, and 10-day exponential moving average of (DJIA). Technical indicators seem more predictive than other world indices, commodity, exchange rate, futures, etc.

## 3.5 Summary

We introduced an R package to implement the ProSGPV algorithm, a variable selection algorithm that incorporates estimation uncertainty via confidence intervals. This novel addition often leads to better support recovery, parameter estimation, and prediction performance. The package is user-friendly, has nice visualization tools for the variable selection process, facilitates subsequent inference and prediction, and very fast computationally. The efficacy of our package is demonstrated on both simulation and real-world data sets.

## References


Albert, A. and Anderson, J. A. (1984). On the existence of maximum likelihood estimates in logistic regression models. *Biometrika*, 71(1):1–10.

Avella-Medina, M. and Ronchetti, E. (2018). Robust and consistent variable selection in high-dimensional generalized linear models. *Biometrika*, 105(1):31–44.

Beretta, A. and Heuchenne, C. (2021). penphcure: Variable selection in proportional hazards cure model with time-varying covariates. *R Journal*, 13(1).

Blume, J. D., D'Agostino McGowan, L., Dupont, W. D., and Greevy Jr, R. A. (2018). Second-generation p-values: Improved rigor, reproducibility, & transparency in statistical analyses. *PLoS One*, 13(3):e0188299.

Blume, J. D., Greevy, R. A., Welty, V. F., Smith, J. R., and Dupont, W. D. (2019). An introduction to second-generation p-values. *The American Statistician*, 73(sup1):157–167.

Bogdan, M., Van Den Berg, E., Sabatti, C., Su, W., and Candès, E. J. (2015). Slope—adaptive variable selection via convex optimization. *The annals of applied statistics*, 9(3):1103.

Cox, D. R. (1972). Regression models and life-tables. *Journal of the Royal Statistical Society: Series B (Methodological)*, 34(2):187–202.

Di Gangi, L., Lapucci, M., Schoen, F., and Sortino, A. (2019). An efficient optimization approach for best subset selection in linear regression, with application to model selection and fitting in autoregressive time-series. *Computational Optimization and Applications*, 74(3):919–948.

Dua, D. and Graff, C. (2017). UCI machine learning repository.

Dvorzak, M. and Wagner, H. (2016). Sparse bayesian modelling of underreported count data. *Statistical Modelling*, 16(1):24–46.

Efroymson, M. (1966). Stepwise regression–a backward and forward look. *Florham Park, New Jersey*.

Fan, J. and Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360.

Fan, J. and Li, R. (2006). Statistical challenges with high dimensionality: Feature selection in knowledge discovery. *arXiv preprint math/0602133*.

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

Fan, J. and Lv, J. (2011). Nonconcave penalized likelihood with np-dimensionality. *IEEE Transactions on Information Theory*, 57(8):5467–5484.

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

Fan, Y. and Lv, J. (2013). Asymptotic equivalence of regularization methods in thresholded parameter space. *Journal of the American Statistical Association*, 108(503):1044–1061.

Fan, Y. and Tang, C. Y. (2013). Tuning parameter selection in high dimensional penalized likelihood. *Journal of the Royal Statistical Society: SERIES B: Statistical Methodology*, pages 531–552.

Fox, J. and Monette, G. (1992). Generalized collinearity diagnostics. *Journal of the American Statistical Association*, 87(417):178–183.

Friedman, J., Hastie, T., and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1.

Gao, Z., Stoev, S., et al. (2020). Fundamental limits of exact support recovery in high dimensions. *Bernoulli*, 26(4):2605–2638.

Giacobino, C., Sardy, S., Diaz-Rodriguez, J., Hengartner, N., et al. (2017). Quantile universal threshold. *Electronic Journal of Statistics*, 11(2):4701–4722.

Harrell Jr, F. E. (2015). *Regression modeling strategies: with applications to linear models, logistic and ordinal regression, and survival analysis*. Springer.

Hastie, T., Tibshirani, R., Tibshirani, R., et al. (2020). Best subset, forward stepwise or lasso? analysis and recommendations based on extensive comparisons. *Statistical Science*, 35(4):579–592.

Hazimeh, H. and Mazumder, R. (2020). Fast best subset selection: Coordinate descent and local combinatorial optimization algorithms. *Operations Research*, 68(5):1517–1537.

Heinze, G., Wallisch, C., and Dunkler, D. (2018). Variable selection–a review and recommendations for the practicing statistician. *Biometrical journal*, 60(3):431–449.

Hoseinzade, E. and Haratizadeh, S. (2019). Cnnpred: Cnn-based stock market prediction using a diverse set of variables. *Expert Systems with Applications*, 129:273–285.

Janson, L., Fithian, W., and Hastie, T. J. (2015). Effective degrees of freedom: a flawed metaphor. *Biometrika*, 102(2):479–485.

Jiang, Y., He, Y., and Zhang, H. (2016). Variable selection with prior information for generalized linear models via the prior lasso method. *Journal of the American Statistical Association*, 111(513):355–376.

Johnson, K. D., Lin, D., Ungar, L. H., Foster, D. P., and Stine, R. A. (2015). A risk ratio comparison of $l\_0$ and $l\_1$ penalized regression. *arXiv preprint arXiv:1510.06319*.

Kaufman, S. and Rosset, S. (2014). When does more regularization imply fewer degrees of freedom? sufficient conditions and counterexamples. *Biometrika*, 101(4):771–784.

Knight, K. and Fu, W. (2000). Asymptotics for lasso-type estimators. *Annals of statistics*, pages 1356–1378.

Kosmidis, I. and Firth, D. (2021). Jeffreys-prior penalty, finiteness and shrinkage in binomial-response generalized linear models. *Biometrika*, 108(1):71–82.

Kozbur, D. (2018). Sharp convergence rates for forward regression in high-dimensional sparse linear models. *University of Zurich, Department of Economics, Working Paper*, 1(253).

Leng, C., Lin, Y., and Wahba, G. (2006). A note on the lasso and related procedures in model selection. *Statistica Sinica*, pages 1273–1284.

Loh, P.-L. and Wainwright, M. J. (2015). Regularized m-estimators with nonconvexity: Statistical and algorithmic theory for local optima. *The Journal of Machine Learning Research*, 16(1):559–616.

Loh, P.-L. and Wainwright, M. J. (2017). Support recovery without incoherence: A case for nonconvex regularization. *The Annals of Statistics*, 45(6):2455–2482.

Ma, R., Tony Cai, T., and Li, H. (2020). Global and simultaneous hypothesis testing for high-dimensional logistic regression models. *Journal of the American Statistical Association*, pages 1–15.

Meier, L., Van De Geer, S., and Bühlmann, P. (2008). The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(1):53–71.

Meinshausen, N. (2007). Relaxed lasso. *Computational Statistics & Data Analysis*, 52(1):374–393.

Meinshausen, N., Bühlmann, P., et al. (2006). High-dimensional graphs and variable selection with the lasso. *The annals of statistics*, 34(3):1436–1462.

Meinshausen, N., Yu, B., et al. (2009). Lasso-type recovery of sparse representations for high-dimensional data. *The annals of statistics*, 37(1):246–270.

Natarajan, B. K. (1995). Sparse approximate solutions to linear systems. *SIAM journal on computing*, 24(2):227–234.

Nelder, J. A. and Wedderburn, R. W. (1972). Generalized linear models. *Journal of the Royal Statistical Society: Series A (General)*, 135(3):370–384.

Rafiei, M. H. and Adeli, H. (2016). A novel machine learning model for estimation of sale prices of real estate units. *Journal of Construction Engineering and Management*, 142(2):04015066.

Rahman, M. S. and Sultana, M. (2017). Performance of firth-and logf-type penalized methods in risk prediction for small or sparse binary data. *BMC medical research methodology*, 17(1):1–15.

Reid, S. and Tibshirani, R. (2014). Regularization paths for conditional logistic regression: the clogitl1 package. *Journal of statistical software*, 58(12).

Saldana, D. F. and Feng, Y. (2018). Sis: An r package for sure independence screening in ultrahigh-dimensional statistical models. *Journal of Statistical Software*, 83(1):1–25.

Salehi, F., Abbasi, E., and Hassibi, B. (2019). The impact of regularization on high-dimensional logistic regression. *arXiv preprint arXiv:1906.03761*.

Shmueli, G. et al. (2010). To explain or to predict? *Statistical science*, 25(3):289–310.

Shortreed, S. M. and Ertefaie, A. (2017). Outcome-adaptive lasso: Variable selection for causal inference. *Biometrics*, 73(4):1111–1122.

Simon, N., Friedman, J., Hastie, T., and Tibshirani, R. (2011). Regularization paths for cox's proportional hazards model via coordinate descent. *Journal of statistical software*, 39(5):1.

Sun, Q., Jiang, B., Zhu, H., and Ibrahim, J. G. (2019). Hard thresholding regression. *Scandinavian Journal of Statistics*, 46(1):314–328.

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

Tibshirani, R. (1997). The lasso method for variable selection in the cox model. *Statistics in medicine*, 16(4):385–395.

van de Geer, S., Bühlmann, P., Zhou, S., et al. (2011). The adaptive and the thresholded lasso for potentially misspecified models (and a lower bound for the lasso). *Electronic Journal of Statistics*, 5:688–749.

Wainwright, M. J. (2009a). Information-theoretic limits on sparsity recovery in the high-dimensional and noisy setting. *IEEE Transactions on Information Theory*, 55(12):5728–5741.

Wainwright, M. J. (2009b). Sharp thresholds for high-dimensional and noisy sparsity recovery using $\ell_1$-constrained quadratic programming (lasso). *IEEE transactions on information theory*, 55(5):2183–2202.

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

Wang, L., Kim, Y., and Li, R. (2013). Calibrating non-convex penalized regression in ultra-high dimension. *Annals of statistics*, 41(5):2505.

Wang, L., You, Y., and Lian, H. (2015). Convergence and sparsity of lasso and group lasso in high-dimensional generalized linear models. *Statistical Papers*, 56(3):819–828.

Wang, M., Song, L., and Wang, X. (2010). Bridge estimation for generalized linear models with a diverging number of parameters. *Statistics & probability letters*, 80(21-22):1584–1596.

Wang, S., Weng, H., Maleki, A., et al. (2020). Which bridge estimator is the best for variable selection? *Annals of Statistics*, 48(5):2791–2823.

Wang, X. and Wang, M. (2016). Variable selection for high-dimensional generalized linear models with the weighted elastic-net procedure. *Journal of Applied Statistics*, 43(5):796–809.

Wasserman, L. and Roeder, K. (2009). High dimensional variable selection. *Annals of statistics*, 37(5A):2178.

Wen, C., Zhang, A., Quan, S., and Wang, X. (2020). Bess: An r package for best subset selection in linear, logistic and cox proportional hazards models. *Journal of Statistical Software*, 94(1):1–24.

Weng, H., Feng, Y., and Qiao, X. (2019). Regularization after retention in ultrahigh dimensional linear regression models. *Statistica Sinica*, 29(1):387–407.

Zhang, C.-H. et al. (2010). Nearly unbiased variable selection under minimax concave penalty. *The Annals of statistics*, 38(2):894–942.

Zhang, T. et al. (2009). Some sharp performance bounds for least squares regression with l1 regularization. *The Annals of Statistics*, 37(5A):2109–2144.

Zhao, P. and Yu, B. (2006). On model selection consistency of lasso. *Journal of Machine learning research*, 7(Nov):2541–2563.

Zheng, Z., Fan, Y., and Lv, J. (2014). High dimensional thresholded regression and shrinkage effect. *Journal of the Royal Statistical Society: Series B: Statistical Methodology*, pages 627–649.

Zhou, S. (2009). Thresholding procedures for high dimensional variable selection and statistical estimation. *Advances in Neural Information Processing Systems*, 22:2304–2312.

Zhou, S. (2010). Thresholded lasso for high dimensional variable selection and statistical estimation. *arXiv preprint arXiv:1002.1583*.

Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American statistical association*, 101(476):1418–1429.

Zuo, Y., Stewart, T. G., and Blume, J. D. (2021a). Variable selection in glm and cox models with second-generation p-values. *arXiv preprint arXiv:2109.09851*.

Zuo, Y., Stewart, T. G., and Blume, J. D. (2021b). Variable selection with second-generation p-values. *The American Statistician*, pages 1–11.