Selecting a Tuning Parameter for the Smoothly Clipped Absolute Deviation Method

Marija Pajdakovska and Jacob Szlachetka

March 2023

Contents

1 Introduction **3**
1.1 Objective 3
1.2 The SCAD Method vs. Existing Methods 3

2 The SCAD Method **4**
2.1 The SCAD Penalty 4
2.2 Selecting a Tuning Parameter (GCV vs. BIC) 5

3 Simulation **6**
3.1 Simulation Setting 6
3.2 Wang, Li and Tsai's Simulation and Remarks 7
3.3 Our Simulation and Remarks 9

4 Conclusion **9**

References **10**

1 Introduction

1.1 Objective

The aim of this project is to present and report on a paper titled "Tuning parameter selectors for the smoothly clipped absolute deviation method" written by Hansheng Wang, Runze Li, and Chih-Ling Tsai. The primary objective of the first section is to provide an explanation of the authors' motivation behind developing this method and to emphasize its advantages over existing methods. In Section 2, this project will delve into a detailed procedure of how the method operates. Additionally, we will present the results of simulation studies concerning methods of tuning parameter selection and draw conclusions based on these results in the final section.

1.2 The SCAD Method vs. Existing Methods

The smoothly clipped absolute deviation (SCAD) method is a statistical technique used for variable selection and estimating parameter coefficients in regression analysis. It was developed by Jianqing Fan and Runze Li in 2001 as a modification of the LASSO (Least Absolute Shrinkage and Selection Operator) method. The LASSO method is a popular approach in machine learning that has several advantages. The objective of LASSO is to minimize the residual sum of squares (RSS) subject to the following penalty $||Y - X\beta||^2 + \lambda \sum_{j=1}^{d} |\beta_j|$, where λ is a tuning parameter, selected through an appropriate method such as cross-validation. One of its key benefits is its ability to perform variable selection by shrinking the coefficients of less important predictors to zero, which can help reduce the complexity of the model and improve its interpretability. Additionally, the LASSO method can handle high dimensional data where the number of predictors is much larger than the number of observations (in the case where $d \geq n$), and it can prevent overfitting by imposing a penalty on the size of the coefficients, which can lead to a more generalized model. The LASSO method can also handle collinearity between features, where two or more features are highly correlated with each other, by selecting only one of them. However, the problem with LASSO lies in the fact that when LASSO penalty is too strong or when there are too many predictors relative to the amount of data, LASSO can underfit the data by forcing the coefficients to be too small or even zero, which may lead to an oversimplified model. The resulting model can miss important patterns in the data and provide severely biased predictions. When the LASSO penalty is too weak or when there are too few predictors relative to the amount of data, LASSO can overfit the data by allowing the model to capture random noise or irrelevant features in the data. This can reduce the efficiency of parameter estimates and predictions. Therefore, obtaining a model that is both parsimonious and predictive is crucial for avoiding bias and improving efficacy. According to Wang, Li, and Tsai (2007), the SCAD method was designed to overcome some of the limitations of the LASSO method, particularly its tendency to produce biased coefficient estimates and to select too many variables when dealing with high-dimensional data. The SCAD method achieves this by replacing the penalty term to the objective function used in the LASSO method. As per Wang, Li, and Tsai (2007) the SCAD method employs a non-convex penalty function that penalizes coefficients more heavily when they are far from zero but less heavily when they are close to zero. This allows the SCAD method to identify important predictors that have small but non-zero coefficients while shrinking coefficients that are not important to zero. The SCAD method can lead to sparser models than LASSO, which is beneficial when there are numerous irrelevant predictors.

Ridge regression also has has several advantages. It is similar in form to LASSO, where the minimization problem is now defined by $||Y - X\beta||^2 + \lambda \sum_{j=1}^{d} \beta_j^2$. It is effective in handling multicollinearity, which can

lead to unstable and unreliable regression coefficients. Similar to LASSO, ridge regression prevents overfitting by adding a penalty term to the sum of squares of the regression coefficients, which shrinks the coefficients towards zero. This makes the model less complex and leads to better generalization performance. Ridge regression is particularly useful when the number of observations is small relative to the number of predictors, as it can help stabilize the estimates of the regression coefficients. Furthermore, ridge regression is easy to implement and computationally efficient, making it a popular choice for many applications. In the comparison between ridge regression and the SCAD method, ridge can reduce the variance of the model, but it does so by increasing its bias, whereas the SCAD penalty does not have the bias that is present in the ridge regression penalty. Additionally, ridge regression does not perform variable selection, meaning that it does not eliminate irrelevant or redundant features from the model, unlike the SCAD method. The penalty term in the SCAD method is a smooth function that is zero when the absolute value of the coefficient estimate is below a threshold, and linear otherwise, as per Wang, Li, and Tsai (2007). This smoothing property allows the SCAD method to produce coefficient estimates that are nearly unbiased, even when the number of variables is large. Overall, the SCAD method has some advantages over LASSO and ridge regression in terms of flexibility and accuracy in identifying important predictors. However, the choice of regularization method ultimately depends on the specific data and modeling goals.

The SCAD method for penalized least squares regression has been demonstrated by Wang, Li, and Tsai (2007) as an effective method for shrinkage and variable selection. This method automatically selects important variables and produces estimators that are as efficient as the oracle estimator (true model). However, the performance of the method relies on selecting the appropriate tuning parameter. As per the study by Wang, Li, and Tsai (2007), existing methods, such as the generalized cross-validation (GCV), may result in overfitting and unsatisfactory tuning parameter selection. To address this issue, we will observe an alternative method for selecting the tuning parameter using the Bayesian information criterion (BIC). The proposed method consistently identifies the true model, as supported by simulation studies. More detailed information on these tuning parameter selectors will be presented in the following section.

2 The SCAD Method

2.1 The SCAD Penalty

Consider the standard linear regression model,

$$y_i = x_i^T \beta + \epsilon_i$$

where y_i is the response variable of the i^{th} observation, x_i is the d-dimentional vector of explanitory variables, β is the d-dimentional vector of the regression coefficients and ϵ_i are the random errors which are i.i.d and normally distributed with mean 0 and variance σ^2. We will denote (x_i, y_i), where $i \in 1, 2, ..., n$, as a random sample. As per Wang, Li, and Tsai (2007), the SCAD method is able to simultaneously select variables and estimate parameters by minimizing the following penalized least squares function:

$$\frac{1}{2n}||Y - X\beta||^2 + \sum_{j=1}^{d} p_\lambda(|\beta_j|) \tag{1}$$

In the above expression, Y denotes the vector $(y_1, ..., y_n)^T$, X represents the vector $(x_1, ..., x_n)^T$, $||\ \ ||$ denotes the Euclidean norm, and the function $p_\lambda(\theta)$, $\theta \in \mathbb{R}$, refers to the smoothly clipped absolute deviation penalty with a tuning parameter $\lambda \in [0, a]$ where $a \in \mathbb{R}^+$, which is typically selected using a data-driven method. It is worth noting that the penalty function satisfies the condition that $p_\lambda(0) = 0$, and its first-order derivative, as given by Wang, Li, and Tsai (2007), is

$$\frac{dp_\lambda}{d\theta} = \lambda[I(\theta \leq \lambda) + \frac{(\alpha\lambda - \theta)_+}{(\alpha - 1)\lambda} I(\theta > \lambda)]$$

where α is taken to be 3.7 by convention. The function $(t)_+$, defined as $tI(t > 0)$, represents the hinge loss function. The estimator obtained by minimizing equation (1) with respect to a given tuning parameter is denoted by $\hat{\beta}_\lambda = (\hat{\beta}_{\lambda 1}, ..., \hat{\beta}_{\lambda d})^T$.

2.2 Selecting a Tuning Parameter (GCV vs. BIC)

According to Wang, Li, and Tsai (2007)'s findings, if λ approaches zero and $n\sqrt{\lambda}$ approaches infinity as the sample size n grows to infinity, the SCAD method can detect irrelevant variables consistently, by producing the respective regression coefficient solutions to zero. Moreover, the SCAD method can identify the true model and provide unbiased estimates of the model parameters, as if it had access to the true underlying distribution of the data, which is referred to as the oracle property. Therefore, it is very important to select an appropriate tuning parameter λ. Typically, the value for the tuning parameter is chosen by minimizing the generalized cross-validation criterion displayed below, given by Wang, Li, and Tsai (2007) as,

$$GCV_\lambda = \frac{||Y - X\hat{\beta}_\lambda||^2}{n(1 - \frac{DF_\lambda}{n})^2} \tag{2}$$

$$= \frac{\hat{\sigma}_\lambda^2}{(1 - \frac{DF_\lambda}{n})^2}$$

where $\hat{\sigma_\lambda}^2 = \frac{||Y - X\hat{\beta}_\lambda||^2}{n}$ and DF_λ is the generalized degrees of freedom, given by Wang, Li, and Tsai (2007) as

$$DF_\lambda = tr(X(X^TX + n\Sigma_\lambda)^{-1}X^T) \tag{3}$$

and $\Sigma_\lambda = diag\{p'_\lambda(|\hat{\beta}_{\lambda 1}|)/|\hat{\beta}_{\lambda_1}|, ..., p'_\lambda(|\hat{\beta}_{\lambda d}|)/|\hat{\beta}_{\lambda d}|\}$, where the entries in this diagonal matrix are polynomial estimates, specifically quadratic approximations to the coefficients in the SCAD penalty function. The optimal tuning parameter then becomes $\hat{\lambda}_{GCV} = argmin_\lambda GCV_\lambda$.

An approximation of the log-transformed GCV_λ can be obtained by the following, as per Wang, Li, and Tsai (2007)

$$log(GCV_\lambda) = log(\hat{\theta}_\lambda^2) - 2log(\frac{1 - DF_\lambda}{n}) \tag{4}$$

$$\approx log(\hat{\theta}_\lambda^2) + \frac{2DF_\lambda}{n}$$

$$\triangleq AIC_\lambda$$

Wang, Li, and Tsai (2007) have shown that GCV_λ closely resembles AIC, the traditional model selection criterion. Although AIC effectively selects the best candidate model of finite dimensions, it is not a consistent

selection criterion since it fails to identify the correct model with a probability approaching 1 in very large samples, where the true model is of finite dimension. Hence, it is possible that the model which $\hat{\lambda}_{GCV}$ selects does not consistently identify the true model of finite dimensions. BIC is known for its consistency, meaning that as the sample size approaches infinity, the probability of selecting the correct model converges to one. This is because BIC has a penalty term that accounts for the number of parameters, preventing overfitting of the data. The paper presented by Wang, Li, and Tsai (2007) in the third section contains detailed proofs of the consistency of BIC, as well as the tendency for GCV to overfit and select irrelevant predictors, which will not be included in this report for the sake of brevity. For BIC, we determine the optimal λ by minimizing the equation below and denoting the result as $\hat{\lambda}_{BIC}$, as given by Wang, Li, and Tsai (2007)

$$BIC_\lambda = log(\hat{\sigma}^2_\lambda) + \frac{DF_\lambda log(n)}{n} \tag{5}$$

The SCAD method in combination with the BIC tuning parameter selector aims to achieve a balance between model accuracy and model complexity (Wang, Li, and Tsai (2007)). The BIC criterion places a heavier penalty on the number of parameters than the GCV criterion, which tends to favor more complex models. This makes the BIC a better choice when the goal is to select a simpler model that is still highly accurate, especially if there is access to a large sample size. We will prove this result with our simulation studies.

3 Simulation

3.1 Simulation Setting

The simulation we conducted is based on a regular linear regression model, which is expressed as $y = x^T\beta + \sigma_\epsilon\epsilon$. To test two different scenarios, we scaled the variance of the model by $\sigma_\epsilon = 1, 3$. In order to compare our simulation to the true model and assess the performance of the chosen method, we established a true beta vector, $\beta = (3, 1.5, 0, 0, 2, 0, 0, 0)^T$. This means that out of the full set of eight predictors, only three are relevant to the response, while the remaining five have no impact. Consequently, the true linear model can be rewritten as $y = \beta_1 x_1 + \beta_2 x_2 + \beta_5 x_5 + \epsilon$.

To generate the predictor space, we used a multivariate normal distribution, $x \sim N_8(0, \Sigma_x)$, where Σ_x is the covariance matrix defined by Wang, Li, and Tsai (2007) as $(\Sigma_x)_{ij} = \rho^{|i-j|}$ for all $i, j = 1, ..., 8$, where $\rho = 0.5$. Although three different ρ values are mentioned in the study by Wang, Li, and Tsai (2007), we only focused on one value of ρ (0.5) in this simulation, since the patterns observed in the final results does not depend on this value. Our main interest was examining the effects of adjusting the sample size and noise level of the model.

We are able to simulate the matching y values given the predictor space to form coordinates for our training set, as we have knowledge of the distribution of ϵ (which is not an assumption). The error is known to be standard normal, so by applying elementary expected value and variance properties to X and Y random variables, where a and b are real numbers:

$E(a) = a$
$E(aX + bY) = aE(X) + bE(Y)$
$V(aX + b) = a^2V(X)$

We can arrive at the following equivalent setting:

$y = x^T\beta + \epsilon$
$\epsilon \sim N(0, \sigma^2)$, where $\sigma = 1, 3$
$y_i \sim N(\beta_1 x_{i1} + \beta_2 x_{i2} + \beta_5 x_{i5}, \sigma^2)$, where $i = 1, ..., n$

In addition, we test for three different sample sizes: $n = 50, 100, 200$, and two different variances, $\sigma = 1, 3$, corresponding to a variance of the model of 1 and 9, respectively.

To estimate the linear regression model, the two methods being GCV and BIC are considered, and the results obtained from these methods are compared with the oracle estimator, which is the OLS estimate of the true model that includes only the relevant predictors. The GCV method is applied using the built-in function cv.ncvreg(), which fits the model with the SCAD penalty and selects the lambda value using cross-validation with 10 folds. However, the BIC method needs to be tested manually, so the corresponding code is provided from scratch. Initially, the interval for possible lambda values to test in the minimization problem is set to $(0, 1]$ with increments of 0.01, covering roughly the same space as the optimal lambda chosen by GCV. After calculating all the required components for computing BIC, the lambda value from this interval that yields the lowest BIC value is selected. While computing the training error is straightforward, computing the degrees of freedom (DF) is much more complicated due to a special case that needs to be considered. To calculate DF, the diagonal matrix Σ_λ, which contains derivatives of the SCAD penalty with estimates for parameter coefficients, is involved. The issue at hand is that the absolute value components from the predicted beta vector are frequently set to 0 as a result of model selection that decides to omit or keep certain predictors. This leads to undefined entries in the matrix, so to resolve this issue, the corresponding row and column belonging to these entries of the diagonal matrix are removed, and the design matrix is reformed to include the selected predictors as well so that the dimensions match (conformability holds).

After selecting the lambda values through the GCV and BIC methods, we need to evaluate the average performance over time using the relative model error concept. Wang, Li, and Tsai (2007) defines model error by

$$ME(\hat{\beta}) = E(x^T\hat{\beta} - x^T\beta)^2 \tag{6}$$

which focuses on error due to the selected method to form $\hat{\beta}$, as opposed to random error. The model errors calculated cover several $\hat{\beta}$ candidates, formed by the unique methods we are familiar with. These methods are denoted as SF (OLS no model selection); OLS (oracle/OLS with correct model selection); BIC (SCAD with lambda chosen through BIC) and GCV (SCAD with lambda chosen through GCV). We calculate $ME_{SF}, ME_{OLS}, ME_{GCV}$ and ME_{BIC}. The last three of these listed errors are then divided by ME_{SF} to obtain the RME, a ratio between 0 and 1 since ME_{SF} has higher error due to the presence of irrelevant predictors. We calculate these errors for all 1000 simulations and use the median of each error as the $MRME$, a more appropriate measure for inference. We choose the median instead of a single RME from any simulation because the SCAD method is not perfect and may lead to incorrect model selection, resulting in an outlier that can misrepresent the most frequent $MRME$ and RME value. This is clear since we can interpret these functions as random variables themselves.

3.2 Wang, Li and Tsai's Simulation and Remarks

The table below presents the simulation results by Wang, Li, and Tsai (2007) which we can use to compare the performance of two methods (BIC and GCV). Our objective is to determine which method is capable of

identifying the true submodel in a regression problem, by comparing their results to those obtained by the oracle estimator. Therefore, we will assess which method produces results that closely approximate those of the oracle estimator. The table provides proportions of underfitted, correctly fitted, and overfitted models. We see that the model selected by BIC has a higher probability of being correctly fitted (72.7%) compared to the model selected by GCV (25.4%), when $\sigma = 3$ and $n = 200$. For overfitted models, the columns labelled '1', '2', and '≥ 3' indicate the proportions of models with one, two, and more than two irrelevant covariates, respectively. When overfitting occurs, the BIC method tends to include only one irrelevant variable into the model (21.9%), compared to including three or more irrelevant variables (0.9%), which is a desirable quality of this method. On the contrary, the generalized cross-validation approach often includes two or more irrelevant variables with a much higher probability than that of BIC (13.4%). The next column labeled 'I' indicates the average number of the three non-zero coefficients that were incorrectly identified as zero, while the column labeled 'C' indicates the average number of the five zero coefficients that were correctly identified. We can see that the model selected by BIC includes approximately five coefficients which were equal to zero (4.528), whereas the model selected by GCV sets less coefficients equal to 0 (3.3). As for the last column, the MRME for the model selected using BIC (42.12%) quickly approaches to the MRME of the true model (34.45%), especially as the sample size increases, compared to the model selected by GCV (55.18), which stays relatively consistent across the three different sample sizes. The results show that the BIC method outperforms the generalized cross-validation method in correctly identifying the true submodel. As expected, both methods perform better as the signal strength increases, which is reflected by a decrease in σ_ϵ from 3 to 1. However, even when $\sigma_\epsilon = 1$ and $n = 200$, the generalized cross-validation method still tends to overfit, while the BIC method overfits less frequently. These findings are consistent with the theoretical results.

				Correctly	Overfitted(%)			No. of Zeros		MRME
σ_ϵ	n	Method	Underfitted(%)	fitted(%)	1	2	≥ 3	I	C	(%)
3	50	$\hat{\lambda}_{GCV}$	10.9	15.9	24.6	25.8	22.8	0.112	3.263	66.78
		$\hat{\lambda}_{BIC}$	15.5	29.3	29.3	18.4	7.5	0.160	3.929	67.04
		Oracle	0	100	0	0	0	0	5	29.29
	100	$\hat{\lambda}_{GCV}$	0.8	23.1	22.6	29.7	23.8	0 08	3.368	58.15
		$\hat{\lambda}_{BIC}$	1.9	51.8	29.4	13.1	3.8	0 19	4.301	52.10
	100	Oracle	0	100	0	0	0	0	5	33.58
	200	$\hat{\lambda}_{GCV}$	0	22.9	21.5	30.5	25.1	0	3.352	54.47
		$\hat{\lambda}_{BIC}$	0	70.0	16.7	10.9	2.4	0	4.540	43.34
		Oracle	0	100	0	0	0	0	5	34.50
1	50	$\hat{\lambda}_{GCV}$	0	26.0	25.7	31.0	17.3	0	3.567	51.93
		$\hat{\lambda}_{BIC}$	0.1	60.3	20.6	13.9	5.1	0 01	4.356	38.31
		Oracle	0	100	0	0	0	0	5	29.30
	100	$\hat{\lambda}_{GCV}$	0	26.3	27.5	27.5	18.7	0	3.567	50.90
		$\hat{\lambda}_{BIC}$	0	67.9	18.9	9.9	3.3	0	4.509	39.10
		Oracle	0	100	0	0	0	0	5	33.42
	200	$\hat{\lambda}_{GCV}$	0	26.5	26.9	28.9	17.7	0	3.582	49.24
		$\hat{\lambda}_{BIC}$	0	75.7	15.7	7.2	1.4	0	4.656	39.01
		Oracle	0	100	0	0	0	0	5	34.77

Figure 1: Article's Simutaion Results

3.3 Our Simulation and Remarks

Below are the results of our simulation study. We have omitted the columns containing the proportions of underfitting, correctly fitting, and overfitting of the model, since the methodology was not explained in detail in the article by Wang, Li, and Tsai (2007).

			No. of Zeros		MRME
σ_ε	n	Method	I	C	(%)
3	50	$\hat{\lambda}_{GCV}$	0.126	3.375	84.44
		$\hat{\lambda}_{BIC}$	0.409	4.702	82.82
		Oracle	0	5	35.19
	100	$\hat{\lambda}_{GCV}$	0.018	3.822	55.40
		$\hat{\lambda}_{BIC}$	0.275	4.904	58.57
		Oracle	0	5	36.14
	200	$\hat{\lambda}_{GCV}$	0	4.309	45.15
		$\hat{\lambda}_{BIC}$	0.17	4.985	43.99
		Oracle	0	5	34.11
1	50	$\hat{\lambda}_{GCV}$	0	4.433	48.10
		$\hat{\lambda}_{BIC}$	0.253	4.994	46.22
		Oracle	0	5	35.19
	100	$\hat{\lambda}_{GCV}$	0	4.465	44.65
		$\hat{\lambda}_{BIC}$	0.16	5	41.45
		Oracle	0	5	36.14
	200	$\hat{\lambda}_{GCV}$	0	4.475	43.50
		$\hat{\lambda}_{BIC}$	0.073	5	37.02
		Oracle	0	5	34.11

When comparing our simulation results to those provided in the article, we observed a similar pattern with respect to sample size and noise. However, the values in each entry of the table were different, evidently due to the randomness in newly simulated training data. It is important to note that the patterns identified in the article simulation are merely trends and may not always hold true. For instance, in the article simulation, GCV for $\sigma = 1$ resulted in an $MRME$ value that increased from a sample size of 50 to 100, then decreased when n was 200. Similarly, in our simulation, GCV slightly outperformed BIC when $n = 100$ and $\sigma = 3$, likely due to increased noise. It is important to note that model selection is not always perfect, and there is a chance that the correct predictors from the true linear model may not be selected, which may lead to breaks in patterns such as the outlined ones above. Additionally, the relationship between GCV/AIC, BIC, and sample size is complex, and the theorem revolving around BIC and sample size holds true only in cases of an infinite amount of data, which does not guarantee this relationship in cases of lower sample size.

4 Conclusion

Overall, we have seen that compared to other popular penalization methods like LASSO and ridge, the SCAD method has been shown to have some advantages. It is a good compromise between sparsity and smoothness, and can lead to better prediction accuracy and variable selection compared to other penalty methods. The

SCAD method provides a good balance between the bias-variance trade off, and is able to effectively handle situations with high-dimensional and correlated data, making it a popular choice in many applications. We have seen in the case of the article's simulation and our own simulation that BIC, as a tuning parameter selector, has the ability to more strongly penalize models with more parameters than GCV, leading to a tendency towards simpler models and better generalization performance. In the simulations, the combination of the SCAD penalty with the BIC produced a model that was comparable to the oracle estimate. The fact that the SCAD penalty paired with BIC was able to produce a model that was similar to the oracle estimate, in the case where sample size is large, suggests that it is a highly effective method for selecting an optimal model.

References

Wang, Hansheng, Runze Li, and Chih-Ling Tsai. 2007. "Tuning Parameter Selectors for the Smoothly Clipped Absolute Deviation Method." *Biometrika*, August. https://doi.org/10.1093/biomet/asm053.