The Square-root Elastic Net With a New Calculating Method

Chen Ai *, Shaoping Shi

Nanchang University Nanchang, China

* Corresponding Author Email: aichen@email.ncu.edu.cn

Abstract. The elastic net is a well-known algorithm and powerful in simultaneous estimation and variable selection under diverse conditions. In this paper, we proposed a novel version called square-root elastic net, which expands the application of the elastic net in some common cases that the noise distribution of data is not normal. We showed a calculating method based on proximal gradient descent combined with backtracking. Similar to the elastic net, the square-root elastic net can select more essential features than the sample size. Furthermore, the mean square error of the square-root elastic net is remarkably reduced when dealing with the abnormal noise compared to the elastic net.

Keywords: Variable selection; Elastic net; Square-root lasso; Proximal gradient.

1. Introduction

There are two fundamental goals of all statistical learning methods: maintaining high prediction accuracy and determining representative features. The frequency of high-dimensional data in our world has changed from rare to common, resulting in the ineffectiveness of traditional statistical methods as the problem is ill-posed.

We consider the most common linear regression model for the *n*-dimensional response **y** given fixed *p*-dimensional regressor matrix **X** first.

$$\boldsymbol{y} = \boldsymbol{X}\beta + \varepsilon \tag{1}$$

where $\boldsymbol{y} = (y_1, \ldots, y_n)^T$ is the response vector, $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p]$ is the regressor matrix and $\mathbf{x}_j = \left(x_{1j}, \ldots, x_{nj}\right)^{\mathrm{T}}$ are linearly independent features. In low-dimensional situations, namely $p < n$, a model fitting procedure produces the vector of coefficients (β, ε). For instance, the ordinary least squares (OLS) estimates with unbiasedness and consistency are obtained by minimizing the residual sum of squares:

$$\beta^* = \min_{\beta} \left\{ \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 \right\} \tag{2}$$

However, as for high-dimensional data, namely $p \gg n$, OLS cannot be used in parameter estimation since the number of solutions of Equation (2) is infinite. Penalization techniques have been proposed to improve OLS due to its shortcomings. The first thought of the penalty term is the L_0 regularization, penalizing models with more covariates but no corresponding improvement in performance. Also, the L_0 regularization results in a balance similar to AIC considering both interpretability and accuracy. Hocking and Leslie [1] propose the original linear regression model with the L0 term, but it cannot obtain the optimal solution owing to non-smoothness and non-convexity.

Among the whole *Lq* regularization family, only the L_1 term is convex. As a result, Tibshirani[2] introduced the L_1 term into the linear regression model for the convenience of calculation and named it LASSO (seen as Equation 3).

$$\beta^* = \min_{\beta} \left\{ \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\} \tag{3}$$

Equation (3) is a convex optimization problem, implying at least one solution. According to Tibshirani[3], the equation may have multiple solutions in the high-dimensional scenario. In addition,

if ε is assumed to obey the Gaussian distribution, then $\beta_{L_1}^*$ can be interpreted as the maximum likelihood estimation of the coefficient penalized on the L_1 term, resulting in sparsity[4].

Admittedly, the lasso estimator is computationally attractive because of its structured convexity, but it can attain the near-oracle performance only when errors are normal and suitable design conditions hold. In other words, the lasso construction relies on knowing the standard deviation of ε. However, when $p \gg n$, estimating the standard deviation of ε is challenging, which remains a complicated theoretical problem.

Fortunately, Belloni [5] proposed the square-root lasso (seen as Equation 4). This method modifies the lasso and matches the lasso's performance with knownεwithout normality or sub-Gaussianity of noise. These results are valid for both Gaussian and non-Gaussian errors, but it happens when the overall number of regressors p is much larger than the sample size n and only s regressors are significant ($s \leq n$).

$$\beta^* = \min_{\beta}\left\{\sqrt{\sum_{i=1}^{n}\left(y_i - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2} + \lambda\sum_{j=1}^{p}|\beta_j|\right\} \tag{4}$$

What if $s > n$ happens? We are inspired by the elastic net [6] (seen as Equation 5) proposed by Zou and Hastie. The lasso selects at most n features before it saturates because of the nature of a convex optimization problem, and thus it is the price of computational convenience. The elastic net introduced the L_2 term on the basis of the lasso, thus having the characteristics of both the lasso and ridge regression[7] and ensuring an effective automatic variable selecting process when $s > n$.

$$\beta^* = \min_{\beta}\left\{\sum_{i=1}^{n}\left(y_i - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2 + \lambda\sum_{j=1}^{p}\left(\alpha|\beta_j| + (1-\alpha)\beta_j^2\right)\right\} \tag{5}$$

In this paper, we propose a new regularization technique which we call the *square-root elastic net* (abbreviated as SREN, seen as Equation 6). This algorithm combines the advantages of the square-root lasso and the elastic net, which can deal with both the situation without a priori information of ε and the data that the number of significant features is more than the sample size. In Section 2, we define the square-root elastic net and its computing method. Empirical results are presented in Section 3, and we discuss the conclusions in Section 4.

2. Square-Root Elastic Net

2.1. Definition and Heuristics

We follow the notation in Section 1 and define the estimation $\widehat{\beta}$ of the coefficient vector β by SREN as the solution of the following convex optimization problem.

$$\hat{\beta} = \min_{\beta}\left\{\sqrt{\sum_{i=1}^{n}\left(y_i - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2} + \lambda\sum_{j=1}^{p}\left[\alpha|\beta_j| + (1-\alpha)\beta_j^2\right]\right\} \tag{6}$$

which can also be written in a simplified matrix form:

$$\hat{\beta} = \min_{\beta}\{\|y - X\beta\|_2 + \lambda[\alpha\|\beta\|_1 + (1-\alpha)\|\beta\|_2^2]\} \tag{7}$$

In the objective function of SREN, despite taking the square root retaining global convexity, the L_1 term is non-differentiable. One most common targeted technique is subgradient because its properties can generally deal with non-differentiable functions, but its convergence rate $O\left(\frac{1}{\sqrt{k}}\right)$ is much slower than that of ordinary gradient descent $O\left(\frac{1}{k}\right)$where k is the number of iterations. Moreover, the definition of subgradient demonstrated that the solution can only make a part of elements in the coefficient vector as close to 0 as possible but cannot acquire a sparse solution, which is not suitable for the SREN.

2.2. Proximal Gradient Descent

We now divide the objective function into two parts and note functions $g(\beta) = \lambda\alpha\|\beta\|_1$ and $f(\beta) = \|y - X\beta\|_2 + \lambda(1-\alpha)\|\beta\|_2^2$, respectively. Both of them are convex, but $f(\beta)$ is differentiable and $g(\beta)$ is non-differentiable. The original optimization problem can be rewritten in a simplified form:

$$\hat{\beta} = \min_{\beta}\{f(\beta) + g(\beta)\} \tag{8}$$

For a quadratic function $f(\beta)$, we can replace $\nabla^2 f(\beta)$ with a constant matrix given known data. By retaining the form of non-differentiable $g(\beta)$, the smooth part of $f(\beta)$ is second-order approximated at β_{k-1} and the recurrence formula of β_k based on a given β_0 is obtained. Let t_k be the step size of *k*-th iteration.

$$\begin{aligned}\min_{\beta}\left\{f(\beta)+g(\beta)\right\} &= \min_{\beta}\left\{f(\beta_{k-1})+\nabla f(\beta_{k-1})^{\mathrm{T}}(\beta-\beta_{k-1})+\frac{1}{2t_k}\|\beta-\beta_{k-1}\|_2^2+g(\beta)\right\}\\ &= \min_{\beta}\left\{\frac{t_k}{2}\|\nabla f(\beta_{k-1})\|_2^2+\nabla f(\beta_{k-1})^{\mathrm{T}}(\beta-\beta_{k-1})+\frac{1}{2t_k}\|\beta-\beta_{k-1}\|_2^2+g(\beta)\right\}\\ &= \min_{\beta}\left\{\frac{1}{2t_k}\|\beta-[\beta_{k-1}-t_k\nabla f(\beta_{k-1})]\|_2^2+g(\beta)\right\}\end{aligned} \tag{9}$$

We define the proximal mapping function $prox_{g,t,k}(\bullet)$ as Equation 10:

$$prox_{g,t,k}\left(\beta_{k-1}-t_k\nabla f(\beta_{k-1})\right) = \min_{\beta}\left\{\frac{1}{2t_k}\|\beta-[\beta_{k-1}-t_k\nabla f(\beta_{k-1})]\|_2^2+g(\beta)\right\} \tag{10}$$

It is obvious that this proximal mapping function is the optimal solution making $\frac{\|\beta-[\beta_{k-1}-t_k\nabla f(\beta_{k-1})]\|_2^2}{2t_k}+g(\beta)$ reach the minimum. The solution must exist because the step size must be positive. In other words, in the case of the non-differentiable $g(\beta)$, $prox_{g,t,k}(\bullet)$ provides a point minimizing the value of $g(\beta)$ and also being close to the non-differentiable point as possible. The nature of $prox_{g,t,k}(\bullet)$ is in line with our expected iterative process, thus:

$$\beta_k = prox_{g,t,k}\left(\beta_{k-1}-t_k\nabla f(\beta_{k-1})\right) \tag{11}$$

The advantage of proximal mapping is that for the non-differentiable part $g(\beta)$ in SREN, its proximal mapping function $prox_{g,t,k}(\bullet)$ has the corresponding analytical solution and, more importantly, is independent of $f(\beta)$. Therefore, the method is applicable for any complex function so long as the gradient of the differentiable part is calculable. By deforming Equation 11, we describe the iterative process in the form of traditional gradient descent, namely proximal gradient descent (PGD), seen as Equation 12:

$$\beta_k = \beta_{k-1} - t_k \cdot G_{t_k}(\beta_{k-1}) \tag{12}$$

where $G_{t_k}(\beta_{k-1}) = \frac{[\beta_{k-1}-prox_{g,t,k}(\beta_{k-1}-t_k\nabla f(\beta_{k-1}))]}{t_k}$.

2.3. Acceleration by Momentum Method

We get inspiration from the idea proposed by Beck and Teboulle[8] that they accelerated the Iterative Shrinkage-Thresholding Algorithm considering momentum and developed the Fast Iterative Shrinkage-Thresholding Algorithm The acceleration carried some momentum from each iteration to reduce the gap between the current direction and the next gradient update direction. According to their settings:

$$\beta_{-1} = \beta_0 \in \mathbb{R}^p \tag{13}$$

$$v_{k-1} = \beta_{k-1} + \frac{k-2}{k+1}(\beta_{k-1} - \beta_{k-2}) \tag{14}$$

We replace β_{k-1} in Equation 11 with v_{k-1} and utilize the properties of the soft-threshold function $\eta_s(w,\lambda) = sgn(w)\,(|w| - \lambda)_+$ to obtain a brand analytical solutions:

$$[\beta_k]_i = \begin{cases} V_i - \lambda\alpha t_k & V_i > \lambda\alpha t_k \\ 0 & |V_i| \le \lambda\alpha t_k \\ V_i + \lambda\alpha t_k & V_i < \lambda\alpha t_k \end{cases} \tag{15}$$

where $[\beta_k]_i$ represents the *i*-th element of the β obtained in the *k*-th iteration, and V_i represents the *i*-th element of $v_{k-1} - t_k \nabla f(v_k)$.

As a result, for a given fixed step size *t*, the accelerated proximal gradient descent (APGD) algorithm satisfies the following inequality:

$$f(\beta_k) - f^* \le \frac{2}{t_k(k+1)^2}\|\beta_0 - \beta^*\|_2^2 \tag{16}$$

In other words, the convergence rate of the APGD algorithm is $O\left(\frac{1}{k^2}\right)$, which is much faster than the traditional gradient descent method. The details of the proof about applications of the momentum method can refer to the papers published by Nesterov [9], Beck and Teboulle [10], and Tseng [11].

3. Empirical Experiment

3.1. Experimental Environment and Parameters

The experimental environment is based on a Windows 10 Pro PC with Intel® Core (TM) i7-8750h CPU @ 2.20GHz and 8GB RAM. The program is based on Python 3.7.6 with mainly used third-party packages including cvxpy 1.1.11, cython 0.29.9, matplotlib 3.1.3, numpy 1.18.1, pandas 1.0.1, scikit-learn 0.22.1, scipy 1.4.1 and sympy 1.5.1.

Observe the form of $\nabla f(\beta)$: In each iteration, for a given β_k, $\nabla f(\beta_k)$ is Lipschitz continuous so there is a constant *L* such that $\|\nabla f(x) - \nabla f(y)\|_2 \le L\|x - y\|_2$. With Taylor expansion, it can be proved that:

$$f(y) \le f(x) + [\nabla f(x)]^T(y - x) + \frac{L}{2}\|y - x\|_2^2 \tag{17}$$

where x and y are different coefficient vectors. If step size $t \in (0, L^{-1})$, then:

$$\begin{aligned} f\left(\beta_k - t_k G_{t_k}(\beta_k)\right) &\le f(\beta_k) - t_k\left[\nabla f(\beta_k)\right]^T G_{t_k}(\beta_k) + \frac{t_k^2 L}{2}\left\|G_{t_k}(\beta_k)\right\|_2^2 \\ &\le f(\beta_k) - t_k\left[\nabla f(\beta_k)\right]^T G_{t_k}(\beta_k) + \frac{t_k}{2}\left\|G_{t_k}(\beta_k)\right\|_2^2 \end{aligned} \tag{18}$$

Based on Formula (18), we use backtracking to search the optimal step size. When $f\left(\beta_k - t_k G_{t_k}(\beta_k)\right) > f(\beta_k) - t_{k-1}[\nabla f(\beta_k)]^T G_{t_k}(\beta_k) + t_k\left\|G_{t_k}(\beta_k)\right\|_2^2/2$, for the initial step size $t_0 = L^{-1}$, we set the step size of each iterations as $t_{k+1} = \eta t_k$, where $\eta \in (0,1)$.

We generate the artificial data with normal noise distribution to inspect the effect of backtracking and the results are shown in Fig 1. It is obvious that the APGD with backtracking can converge much faster.

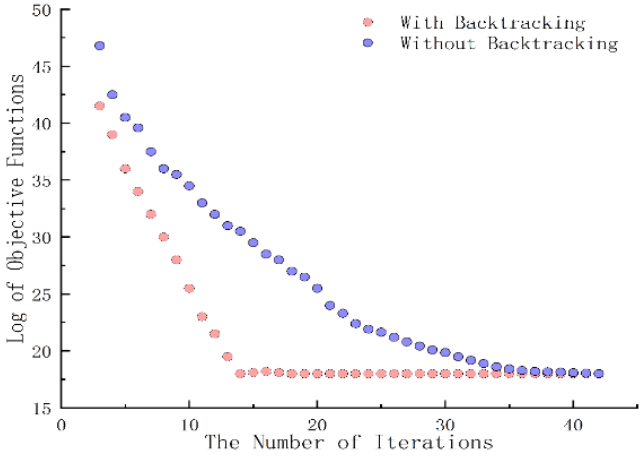

Fig 1. Comparison of convergence speed with and without backtracking

In the following parts, we selected $\lambda = c\sqrt{n} \cdot \Phi^{-1}\left(\frac{1-\iota}{2p}\right)$ given by Belloni[12] to estimate the penalty parameter. In this formula, $c > 1$ is the theoretical constant deduced by Bickel et al.[13], and $1-\iota$ is the confidence level that the estimation approximately achieves the oracle property, referring to Lemma II.1. proved by Bunea et al. [14].

3.2. Experimental Process and Data

The objective function of SREN is a strongly convex function with the convex measure $2\lambda(1\text{-}\alpha)$, hence the optimal solution β can be acquired from any initial β_0. Set β_0 as zero vector, and the overall flow of the algorithm is shown in Fig 2.

ALGOTIRHM 1: square-root elastic net

Data: regressor matrix $X \in \mathbb{R}^{n\times p}$ and response vector $y \in \mathbb{R}^{n\times 1}$

Input: hyperparameter η and α

Output: coefficient $\beta \in \mathbb{R}^{p\times 1}$

Set constant $\lambda = c\sqrt{n} \cdot \Phi^{-1}[\frac{1-\iota}{2p}]$ and $L = \frac{X^\top X}{\|y\|_2^2} + 2\lambda(1-\alpha)$;

Set start point $\beta_0 = \beta_{-1} = \mathbf{0}^{p\times 1}$ and start stepsize $t_0 = L^{-1}$;

while $\beta_k \neq \beta_{k-1} (k \geqslant 1)$ **do**

$v_{k-1} = \frac{2k-1}{k+1}\beta_{k-1} - \frac{k-2}{k+1}\beta_{k-2}$;

if $\Delta f(\beta_k) > \frac{t_{k-1}}{2}[\|G_{t_{k-1}}(\beta_{k-1})\|_2^2 - 2\|\nabla f(v_{k-1})^\top G_{t_{k-1}}(\beta_{k-1})\|_2^2]$

then $t_k = \eta t_{k-1}$;

else $t_k = t_{k-1}$;

$\beta_{k+1} = prox_{g,t,k}(v_k - t_k \nabla f(v_k))$

end

Fig 2. Flow of square-root elastic net algorithm

Note the artificial experimental data as $(y_i, \boldsymbol{x}_i) \in \mathbb{R}^{p+1}$ satisfying:

$$y_i = \boldsymbol{x}_i^\top \beta + \sigma\varepsilon_i \tag{19}$$

where $i \in N^+$, and ε_i are identically and independently distributed.

We use Monte Carlo simulation to generate three groups of experimental data, according to the two characteristics of SREN algorithm.

(1) No priori information of ε_i is required;

(2) The number of essential features exceeding the sample size can be selected.

We take the first m elements as 1 in the coefficient vector β and others as 0, i.e., $\beta = [1,1,\ldots,1,0,0,\ldots,0]^\top$. More properties are shown in Table 1.

Table 1. Description of similarities and differences

PROPERTIES	Control Group	Experimental Group 1	Experimental Group 2
ε_i distribution	$N(0,1)$	$\frac{t(4)}{\sqrt{2}}$	$N(0,1)$
Partition Number m	$0.2n$	$0.2n$	$1.2n$
Feature Number p	Uniformly take $5n$		
Sample Size n	Take 10 or 100: small and large sample sizes		
Noise Rate σ	Take 0.5, 1 or 2: weak, moderate and strong noise scales		
x_i distribution	$x_i \sim N(0, \Sigma)$, where Σ is a Toeplitz matrix[12], $\Sigma_{jk} = \left(\frac{1}{2}\right)^{\lvert j-k \rvert}$		

We compare these two experimental groups with the control group data to examine the characteristics of SREN. Different sample sizes and different noise scales construct six scenarios at all, and we run LASSO, elastic net, square-root lasso, and SREN in each scenario to collect some enlightening results.

3.3. Experimental Results

The mean square errors corresponding to each optimal penalty parameter in different scenarios are exhibited in Table 2, and the results are round to two decimal places. The number within brackets represents the training error, while the number without brackets represents the test error.

We abbreviate the control group as CG, the experimental group 1 as EG1, and the experimental group 2 as EG2 in following parts.

Table 2. Mean square errors in different scenarios

SCENARIOS	Groups	LASSO	Elastic Net	Square-root LASSO	SREN
$n = 10$ $\sigma = 0.5$	CG	0.31(0.29)	0.36(0.27)	0.27(0.32)	0.29(0.30)
	EG1	0.55(0.35)	0.36(0.37)	0.29(0.08)	0.30(0.12)
	EG2	1.42(1.08)	1.29(0.60)	0.60(0.15)	0.51(0.14)
$n = 100$ $\sigma = 0.5$	CG	0.34(0.30)	0.35(0.42)	0.09(0.06)	0.09(0.08)
	EG1	0.99(0.45)	1.01(0.38)	0.15(0.06)	0.14(0.08)
	EG2	1.78(1.62)	1.46(1.24)	0.21(0.08)	0.19(0.12)
$n = 10$ $\sigma = 1$	CG	0.80(0.38)	0.53(0.38)	0.29(0.37)	0.29(0.36)
	EG1	0.89(0.68)	0.84(0.69)	0.41(0.37)	0.41(0.37)
	EG2	5.42(3.96)	3.43(2.65)	0.85(0.86)	0.79(0.77)
$n = 100$ $\sigma = 1$	CG	1.29(0.93)	0.96(0.81)	0.15(0.14)	0.15(0.14)
	EG1	2.11(1.83)	1.82(0.95)	0.22(0.18)	0.22(0.18)
	EG2	4.05(2.76)	3.03(2.17)	0.33(0.21)	0.23(0.11)
$n = 10$ $\sigma = 2$	CG	4.85(3.47)	7.54(1.33)	1.09(0.66)	1.04(0.79)
	EG1	2.23(3.35)	2.45(3.99)	0.78(0.90)	0.78(0.84)
	EG2	7.52(6.21)	7.15(5.94)	0.99(0.81)	0.82(0.57)
$n = 100$ $\sigma = 2$	CG	2.97(0.58)	2.85(2.09)	0.26(0.24)	0.26(0.22)
	EG1	10.15(7.79)	10.97(6.42)	0.51(0.35)	0.50(0.32)
	EG2	6.44(3.57)	5.53(2.74)	0.39(0.31)	0.26(0.28)

The regularization paths are shown from Figure 3 to Figure 8, in which the first row of each figure is the CG, the second row is the EG1 and the third row is the EG2. These paths reveal the variable selection process to observe how various algorithms ultimately pick up essential features under the condition of different penalty coefficients λ.

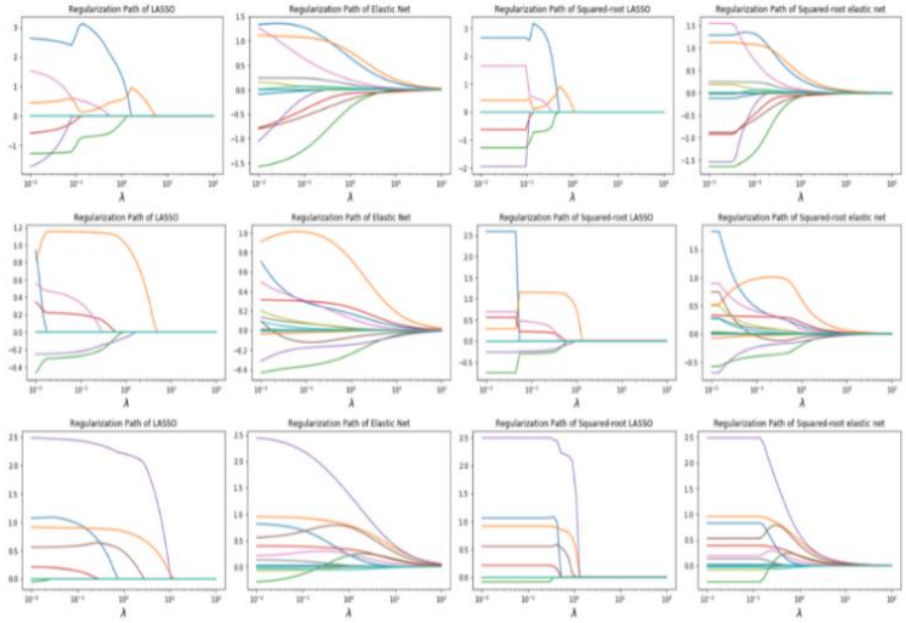

Fig 3. Regularization Path with Small Sample and Weak Noise

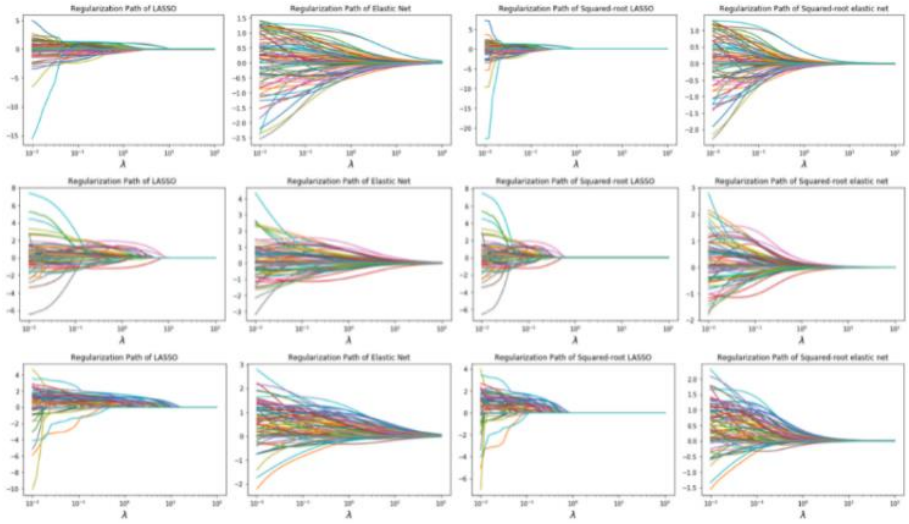

Fig 4. Regularization Path with Small Sample and Weak Noise

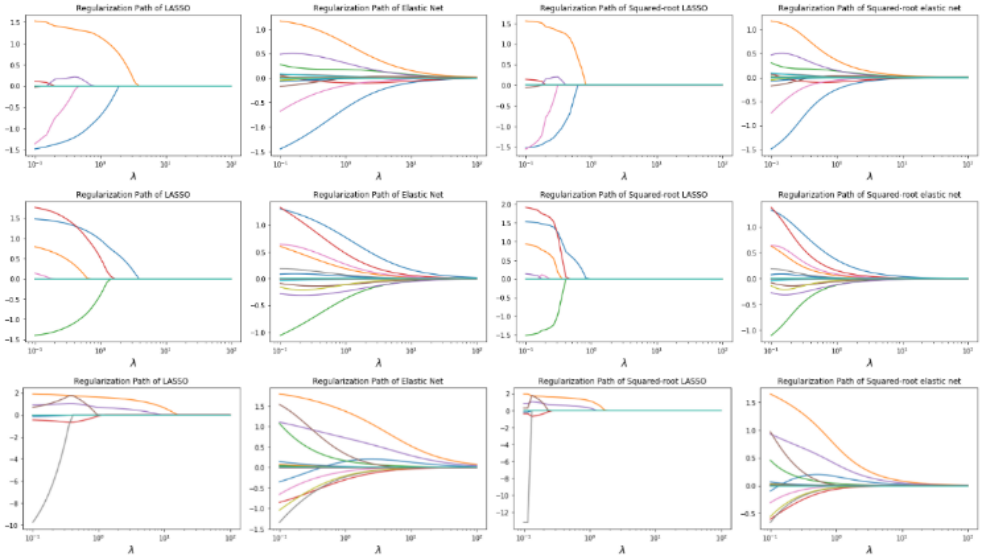

Fig 5. Regularization Path with Small Sample and Weak Noise

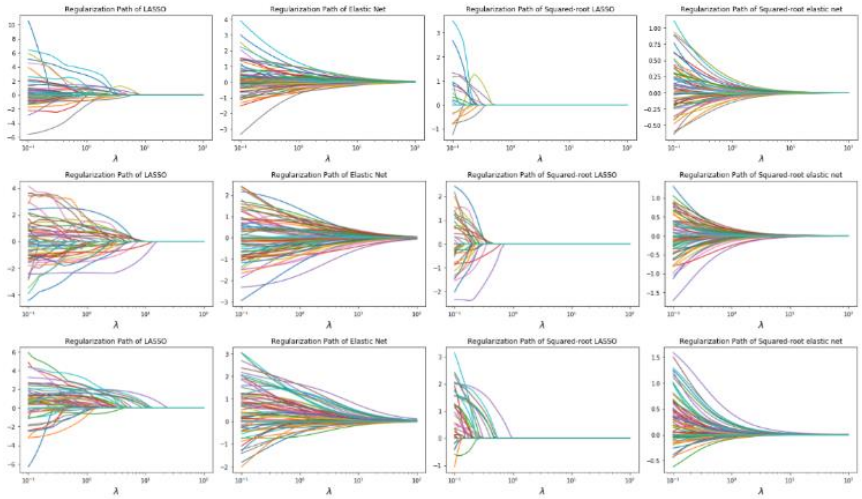

Fig 6. Regularization Path with Large Sample and Moderate Noise

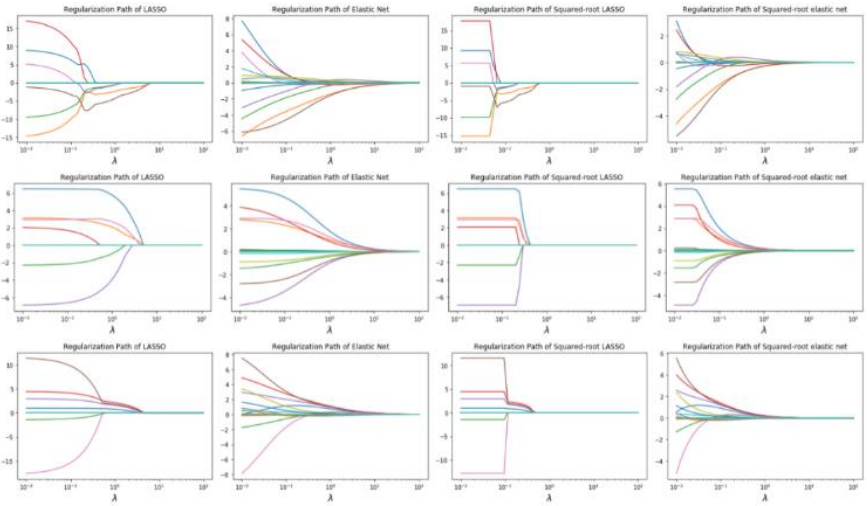

Fig 7. Regularization Path with Small Sample and Strong Noise

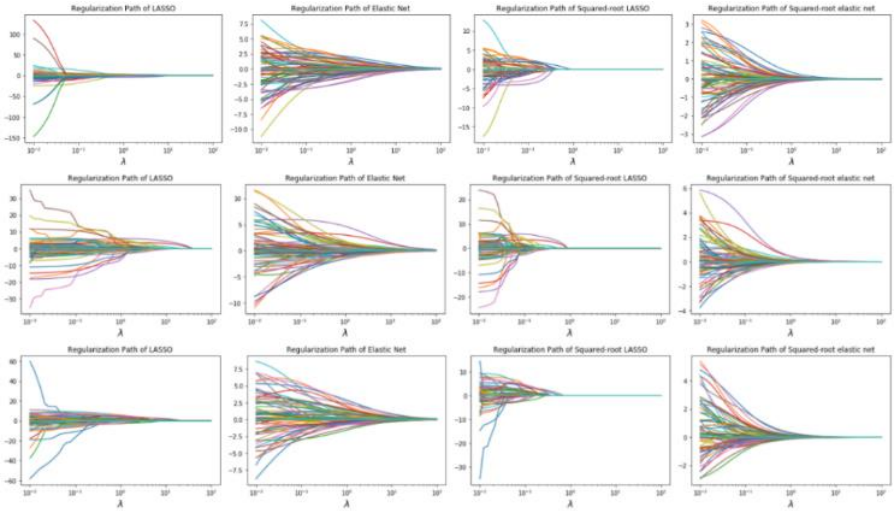

Fig 8. Regularization Path with Large Sample and Strong Noise

4. Conclusion

Concerning mean square error, the performance of square-root lasso and SREN surpasses lasso and elastic net. The advantage increases with the growth of sample size and noise scale. It is worth

noting that the performance of lasso and elastic net in processing data with standard normal noise distribution (CG) is not much different from that of square root lasso and SREN.

However, when dealing with the data with some other noise distributions (EG1), the performance of lasso and the elastic net is not stable. The fluctuation degree is much higher than that of square root lasso and SREN. The amplitude immensely increases with the enlargement of noise.

Moreover, when dealing with the data with more essential features than the sample size (EG2), square root lasso and SREN still have the advantage of approaching mean square error. Nevertheless, in between, the optimal penalty coefficient determined by square root lasso cannot effectively screen out all essential features (Even the number of essential features ignored is always more than half of the total).

Finally, we conclude that SREN is a more comprehensive algorithm because the SREN can competently screen the essential characteristics under various conditions, which can undoubtedly provide necessary assistance for relevant researchers to explore the principle of the problem and strengthen the interpretability of their results.

References

[1] Hocking R. R., Leslie, R. N. Selection of the best subset in regression analysis [J]. Technometrics, 1967, 9(4): 531-540.

[2] Tibshirani R. Regression shrinkage and selection via the lasso [J]. Journal of the Royal Statistical Society, Series B, 1996, 58(1).

[3] Tibshirani, Ryan J. The Lasso Problem and Uniqueness [J]. Electronic Journal of Stats, 2013, 7(1): 1456-1490.

[4] Tibshirani R. Regression shrinkage and selection via the lasso: a retrospective[J]. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 2011, 73(3): 273-282.

[5] Belloni A, Chernozhukov V, Wang L. Square-root lasso: pivotal recovery of sparse signals via conic programming [J]. Biometrika, 2011, 98(4): 791-806.

[6] Zou H, Hastie T. Regularization and variable selection via the elastic net[J]. Journal of the royal statistical society: series B (statistical methodology), 2005, 67(2): 301-320.

[7] Fu W J. Penalized regressions: the bridge versus the lasso[J]. Journal of computational and graphical statistics, 1998, 7(3): 397-416.

[8] Beck A, Teboulle M. Global Optimality Conditions for Quadratic Optimization Problems with Binary Constraints[J]. Siam Journal on Optimization, 2008, 11(1):179-188.

[9] Nesterov Y. Smooth minimization of non-smooth functions [J]. Mathematical Programming, 2005, 103(1):127-152.

[10] Beck A, Tabouleh M. A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems[J]. Siam J Imaging Sciences, 2009, 2(1): 183-202.

[11] Tseng P. On accelerated proximal gradient methods for convex-concave optimization [J]. submitted to SIAM Journal on Optimization, 2008, 2(3).

[12] Belloni A, Chernozhukov V, Wang L. Square-root lasso: pivotal recovery of sparse signals via conic programming [J]. Biometrika, 2011, 98(4): 791-806.

[13] Bickel P J, Ritov Y, Tsybakov A B. Simultaneous analysis of Lasso and Dantzig selector [J]. The Annals of statistics, 2009, 37(4): 1705-1732.

[14] F Bunea, Lederer J, She Y. The Group Square-Root Lasso: Theoretical Properties and Fast Algorithms[J]. IEEE Transactions on Information Theory, 2013, 60(2): 1313-1325.