An R Package`flare` for High Dimensional Linear Regression and Precision Matrix Estimation

Xingguo Li*† Tuo Zhao*‡ Xiaoming Yuan§ Han Liu¶

Abstract

This paper describes an `R` package named `flare`, which implements a family of new high dimensional regression methods (LAD Lasso, SQRT Lasso, ℓ_q Lasso, and Dantzig selector) and their extensions to sparse precision matrix estimation (TIGER and CLIME). These methods exploit different nonsmooth loss functions to gain modeling flexibility, estimation robustness, and tuning insensitiveness. The developed solver is based on the alternating direction method of multipliers (ADMM), which is further accelerated by the multistage screening approach. The package `flare` is coded in double precision `C`, and called from `R` by a friendly user interface. The memory usage is optimized by using the sparse matrix output. The experiments show that `flare` is efficient and can scale up to large problems.

1 Introduction

As a popular sparse linear regression method for high dimensional data analysis, Lasso has been extensively studied by machine learning and statistics communities (Tibshirani, 1996; Chen et al., 1998). It adopts the quadratic loss and ℓ_1 norm regularization functions to select and estimate nonzero parameters simultaneously. Software packages such as `glmnet` have been developed to efficiently solve large problems(Friedman et al., 2010). Lasso further yields a wide range of research interests, and motivates many variants by exploiting nonsmooth loss functions to gain modeling flexibility, estimation robustness, and tuning insensitiveness. These nonsmooth loss functions, however, pose a great challenge to computation. To the best of our knowledge, no efficient solver has been developed so far for these Lasso variants.

In this report, we describe a newly developed `R` package named `flare` (Family of Lasso Regression). the `flare` package implements a family of linear regression methods including LAD Lasso (Wang,

*Xingguo Li and Tuo Zhao contributed equally to this work;
†Department of Mathematics and Statistics, University of Minnesota Duluth;
‡Department of Computer Science, Johns Hopkins University;
§Department of Mathematics, Hong Kong Baptist University;
¶Department of Operations Research and Financial Engineering, Princeton University.

2013), SQRT Lasso (Belloni et al., 2011), ℓ_q Lasso, and Dantzig selector (Candes and Tao, 2007). By adopting the column by column regression scheme, we further extend these regression methods to sparse precision matrix estimation estimation including TIGER (Liu and Wang, 2012) and CLIME (Cai et al., 2011)). The developed solver is based on the alternating direction method of multipliers (ADMM), which is further accelerated by a multistage screening approach (Gabay and Mercier, 1976; Boyd et al., 2011). The global convergence result of ADMM has been established in He and Yuan (2012a,b). The numerical simulations show that the `flare` package is efficient and can scale up to large problems.

2 Notation

We first introduce some notations. Given a d-dimensional vector $\boldsymbol{v} = (v_1, \ldots, v_d)^T \in \mathbb{R}^d$, we define vector norms:

$$||\boldsymbol{v}||_q^q = \sum_j |v_j|^q, \; ||\boldsymbol{v}||_\infty = \max_j |v_i|.$$

where $1 \leq q \leq 2$. Given a matrix $\mathbf{A} = [\mathbf{A}_{jk}] \in \mathbb{R}^{d\times d}$, we use $||\mathbf{A}||_2$ to denote the largest singular value of $\mathbf{A}$. We also define the winterization, univariate soft thresholding, and group soft thresholding operators as follows,

$$\begin{aligned}
\text{Winterization:}\;\; & \mathcal{W}_\lambda(\boldsymbol{v}) = [\text{sign}(v_j)\cdot\min\{|v_j|,\lambda\}]_{j=1}^d,\\
\text{Univariate Soft Thresholding:}\;\; & \mathcal{S}_\lambda(\boldsymbol{v}) \;= [\text{sign}(v_j)\cdot\max\{|v_j|-\lambda,0\}]_{j=1}^d,\\
\text{Group Soft Thresholding:}\;\; & \mathcal{G}_\lambda(\boldsymbol{v}) \;= \frac{\boldsymbol{v}}{||\boldsymbol{v}||_2}\cdot\max\{||v_j||_2-\lambda,0\}.
\end{aligned}$$

3 Algorithm

We are interested in solving convex programs in the following generic form,

$$\widehat{\boldsymbol{\beta}} = \operatorname*{argmin}_{\boldsymbol{\beta},\,\boldsymbol{\alpha}} L_\lambda(\boldsymbol{\alpha}) + \|\boldsymbol{\beta}\|_1 \quad \text{subject to } \boldsymbol{r} - \mathbf{A}\boldsymbol{\beta} = \boldsymbol{\alpha}. \tag{1}$$

where $\lambda > 0$ is the regularization parameter. The possible choices of $L_\lambda(\boldsymbol{\alpha})$, $\mathbf{A}$, and $\boldsymbol{r}$ for different regression methods are listed in Table1. As can be seen, LAD Lasso and SQRT Lasso are special examples of ℓ_q Lasso for $q = 1$ and $q = 2$ respectively. All methods above can be efficiently solved by the iterative scheme as follows,

$$\boldsymbol{\alpha}^{t+1} = \operatorname*{argmin}_{\boldsymbol{\alpha}} \frac{1}{2}\left\|\boldsymbol{u}^t + \boldsymbol{r} - \mathbf{A}\boldsymbol{\beta}^t - \boldsymbol{\alpha}\right\|_2^2 + \frac{1}{\rho}L_\lambda(\boldsymbol{\alpha}), \tag{2}$$

$$\boldsymbol{\beta}^{t+1} = \operatorname*{argmin}_{\boldsymbol{\beta}} \frac{1}{2}\left\|\boldsymbol{u}^t - \boldsymbol{\alpha}^{t+1} + \boldsymbol{r} - \mathbf{A}\boldsymbol{\beta}\right\|_2^2 + \frac{1}{\rho}\|\boldsymbol{\beta}\|_1, \tag{3}$$

$$\boldsymbol{u}^{t+1} = \boldsymbol{u}^t + (\boldsymbol{r} - \boldsymbol{\alpha}^{t+1} - \mathbf{A}\boldsymbol{\beta}^{t+1}), \tag{4}$$

where $\boldsymbol{u}$ is the rescaled Lagrange multiplier, and $\rho > 0$ is the penalty parameter. For LAD Lasso, SQRT Lasso, and Dantzig selector, we can obtained a closed form solution to(2) by

$$\text{LAD Lasso:}\ \ \boldsymbol{\alpha}^{t+1} = \mathcal{S}_{\frac{1}{n\rho\lambda}}(\boldsymbol{u}^t + \boldsymbol{r} - \mathbf{A}\boldsymbol{\beta}^t), \tag{5}$$

$$\text{SQRT Lasso:}\ \ \boldsymbol{\alpha}^{t+1} = \mathcal{G}_{\frac{1}{\sqrt{n}\rho\lambda}}(\boldsymbol{u}^t + \boldsymbol{r} - \mathbf{A}\boldsymbol{\beta}^t), \tag{6}$$

$$\text{Dantzig selector:}\ \ \boldsymbol{\alpha}^{t+1} = \mathcal{W}_{\lambda}(\boldsymbol{u}^t + \boldsymbol{r} - \mathbf{A}\boldsymbol{\beta}^t). \tag{7}$$

For ℓ_q Lasso with $1 < q < 2$, we can solve(2) by the bisection based root finding algorithm (Liu and Ye, 2010).(3) is a standard ℓ_1 penalized least square problem. Our solver adopts the linearization at $\boldsymbol{\beta} = \boldsymbol{\beta}^t$ as follows and solve(3) approximately.

$$\boldsymbol{\beta}^{t+1} = \operatorname*{argmin}_{\boldsymbol{\beta}} \frac{1}{2}\left\|\boldsymbol{\beta} - \boldsymbol{\beta}^t + \mathbf{A}^T(\mathbf{A}\boldsymbol{\beta}^t - \boldsymbol{u}^t + \boldsymbol{\alpha}^{t+1} - \boldsymbol{r})/\gamma\right\|_2^2 + \frac{1}{\gamma\rho}\|\boldsymbol{\beta}\|_1, \tag{8}$$

where $\gamma = ||\mathbf{A}||_2^2$. We can obtain a closed form solution to(8) by soft thresholding,

$$\boldsymbol{\beta}^{t+1} = \mathcal{S}_{\frac{1}{\gamma\rho}}\left(\boldsymbol{\beta}^t - \mathbf{A}^T(\mathbf{A}\boldsymbol{\beta}^t - \boldsymbol{u}^t + \boldsymbol{\alpha}^{t+1} - \boldsymbol{r})/\gamma\right). \tag{9}$$

Beside the pathwise optimization scheme and the active set trick, we also adopt the multistage screening approach to speedup the computation. In particular, we first select k nested subsets of coordinates $\mathcal{A}_1 \subseteq \mathcal{A}_2 \subseteq ... \subseteq \mathcal{A}_k = \mathbb{R}^d$ by the marginal correlation between covariates and response. Then the algorithm iterates over these nested subsets of coordinates to obtain the solution. The multistage screening approach can greatly boost the empirical performance especially for Dantzig selector.

4 Examples

We illustrate the user interface by two examples. The first one is the eye disease dataset in our package.

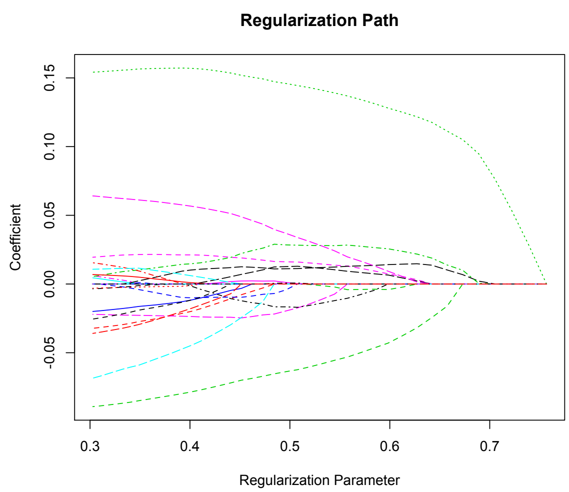
```
> library(flare); data(eyedata)                    # Load the dataset
> out1 = slim(x,y,method="lq",q=2,nlambda=40,lambda.min.ratio=0.5)
> out2 = slim(x,y,method="dantzig",nlambda=40,lambda.min.ratio=0.35)
> plot(out1); plot(out2)                           # Plot solution paths
```

The program automatically generates a sequence of 40 regularization parameters and estimates the corresponding solution paths of SQRT Lasso and Dantzig selector. We further plot two solution paths in Figure 1.

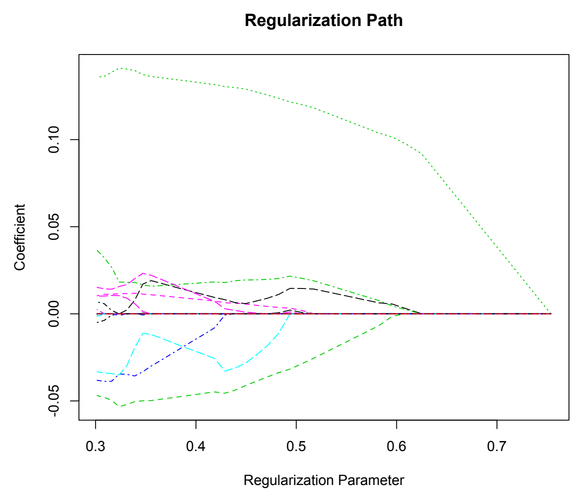
Our second example is the simulated dataset using the data generator in our package.

Table 1: All regression methods provided in the `flare` package. $\mathbf{X} \in \mathbb{R}^{n\times d}$ denotes the design matrix, and $\boldsymbol{y} \in \mathbb{R}^n$ denotes the response vector. "L.P." denotes the general linear programming solver, and "S.O.C.P" denotes the second order cone programming solver.

Method	Loss function	$\mathbf{A}$	$\boldsymbol{b}$	Existing solver
LAD Lasso	$L_\lambda(\boldsymbol{\alpha}) = \frac{1}{n\lambda}\|\boldsymbol{\alpha}\|_1$	$\mathbf{X}$	$\boldsymbol{y}$	L.P.
SQRT Lasso	$L_\lambda(\boldsymbol{\alpha}) = \frac{1}{\sqrt{n}\lambda}\|\boldsymbol{\alpha}\|_2$	$\mathbf{X}$	$\boldsymbol{y}$	S.O.C.P.
ℓ_q Lasso	$L_\lambda(\boldsymbol{\alpha}) = \frac{1}{\sqrt[q]{n}\lambda}\|\boldsymbol{\alpha}\|_q$	$\mathbf{X}$	$\boldsymbol{y}$	None
Dantzig selector	$L_\lambda(\boldsymbol{\alpha}) = \begin{cases} \infty & \text{if } \|\boldsymbol{\alpha}\|_\infty > \lambda \\ 0 & \text{otherwise} \end{cases}$	$\frac{1}{n}\mathbf{X}^T\mathbf{X}$	$\frac{1}{n}\mathbf{X}^T\boldsymbol{y}$	L.P.



(a) SQRT Lasso



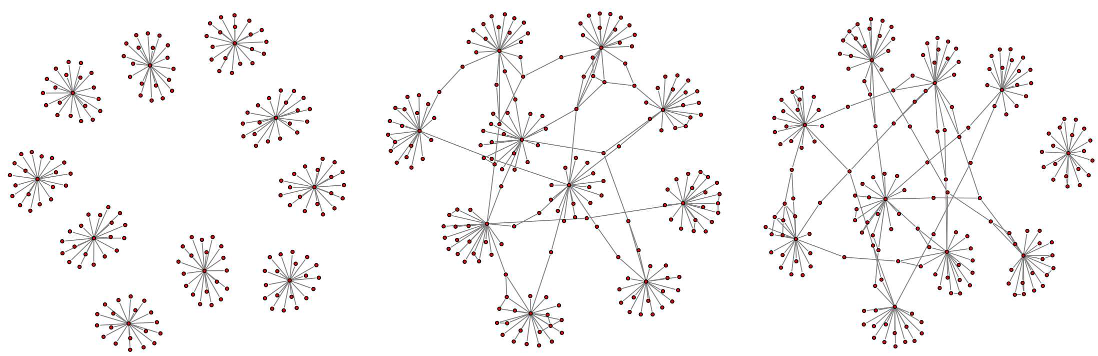
(b) Dantzig selector

Figure 1: Solution paths obtained by the package`flare`.

```
> # Generate data with hub structure
> L = sugm.generator(n=400,d=200,graph="hub",g=10)
> out1 = sugm(L$data,method="clime",nlambda=10,lambda.min.ratio=0.4)
> # Model selection using cross validation.
> out1.opt = sugm.select(out1,criterion="cv")
> out2 = sugm(L$data,lambda = sqrt(log(200)/400))
```

```
> # Visualize obtained grpahs
> plot(L); plot(out1.opt); plot(out2)
```

For CLIME, the program automatically generates a sequence of 10 regularization parameters, estimates the corresponding graph path, and chooses the optimal regularization parameter by cross validation. For TIGER, we manually choose the regularization to be $\sqrt{\log(d)/n}$. We then compare the obtained graphs with the true graph using the visualization functions in our package, and the resulting figures are presented in Figure 2.



(a) Truth (b) CLIME (c) TIGER

Figure 2: Graphs estimated by the package`flare`.

5 Numerical Simulation

All experiments below were carried out on a PC with Intel Core i5 3.3GHz processor, and the convergence threshold of `flare` is chosen to be 10^{-4}. Timings (in seconds) are averaged over 50 replications using a sequence of 20 regularization parameters, and the range of regularization parameters is chosen so that each method produced approximately the same number of nonzero estimates.

We first evaluate the timing performance of `flare` for sparse linear regression. We set $n = 100$ and vary d from 375 to 3000 as is shown in Table 2. We independently generate each row of the design matrix from a d-dimensional normal distribution $N(0, \mathbf{\Sigma})$, where $\mathbf{\Sigma}_{jk} = 0.5^{|j-k|}$. Then we generate the response vector using $y_i = 3\mathbf{X}_{i1} + 2\mathbf{X}_{i2} + 1.5\mathbf{X}_{i4} + \epsilon_i$, where ϵ_i is independently generated from $N(0, 1)$. From Table 2, we see that all methods achieve very good timing performance. Dantzig selector and ℓ_q Lasso are slower due to more difficult computational formulations. For comparison purpose, we also present the timing performance of the `glmnet` package for solving SQRT Lasso in Table 2. Since `glmnet` cannot be directly applied to SQRT Lasso, the implementation is based on the alternating minimization algorithm proposed in Sun and Zhang (2012). In particular, this algorithm obtains the minimizer by solving a sequence of Lasso problems (using `glmnet`). As can be seen, it also achieves good timing performance, but still slower than the `flare` package.

We then evaluate the timing performance of `flare` for sparse precision matrix estimation. We set $n = 100$ and vary d from 100 to 400 as is shown in Table 2. We independently generate the data from a d-dimensional normal distribution $N(0, \mathbf{\Sigma})$, where $\mathbf{\Sigma}_{jk} = 0.5^{|j-k|}$. The corresponding precision matrix $\mathbf{\Omega} = \mathbf{\Sigma}^{-1}$ has $\mathbf{\Omega}_{jj} = 1.3333$, $\mathbf{\Omega}_{jk} = -0.6667$ for all $j, k = 1, ..., d$ and $|j - k| = 1$, and all other entries are 0. As can be seen from Table 2, TIGER and CLIME both achieve good timing performance, and CLIME is slower than TIGER due to a more difficult computational formulation.

Table 2: Average timing performance (in seconds) with standard errors in the parentheses on sparse linear regression and sparse precision matrix estimation.

Sparse Linear Regression				
Method	$d = 375$	$d = 750$	$d = 1500$	$d = 3000$
LAD Lasso	1.1713(0.2915)	1.1046(0.3640)	1.8103(0.2919)	3.1378(0.7753)
$\ell_{1.5}$ Lasso	12.995(0.5535)	14.071(0.5966)	14.382(0.7390)	16.936(0.5696)
Dantzig selector	0.3245(0.1871)	1.5360(1.8566)	4.4669(5.9929)	17.034(23.202)
SQRT Lasso (`flare`)	0.4888(0.0264)	0.7330(0.1234)	0.9485(0.2167)	1.2761(0.1510)
SQRT Lasso (`glmnet`)	0.6417(0.0341)	0.8794(0.0159)	1.1406(0.0440)	2.1675(0.0937)
Sparse Precision Matrix Estimation				
Method	$d = 100$	$d = 200$	$d = 300$	d=400
TIGER	1.0637(0.0361)	4.6251(0.0807)	7.1860(0.0795)	11.085(0.1715)
CLIME	2.5761(0.3807)	20.137(3.2258)	42.882(18.188)	112.50(11.561)

6 Discussion and Conclusions

Though the `glmnet` package cannot handle nonsmooth loss functions, it is much faster than `flare` for solving Lasso as illustrated in Table 3. The simulation setting is the same as the sparse linear regression setting in §5. Moreover, the `glmnet` package can also be applied to solve ℓ_1 regularized generalized linear model estimation problems, which `flare` cannot. Overall speaking, the `flare` package serves as an efficient complement to the `glmnet` packages for high dimensional data analysis. We will continue to maintain and support this package.

Acknowledgement Tuo Zhao and Han Liu are supported by NSF Grants III-1116730 and NSF III-1332109, NIH R01MH102339, NIH R01GM083084, and NIH R01HG06841, and FDA

Table 3: Quantitive comparison between the `flare` and `glmnet` packages for solving Lasso.

Method	$d = 375$	$d = 750$	$d = 1500$	$d = 3000$
Lasso (`flare`)	0.0920(0.0013)	0.1222(0.0009)	0.2328(0.0037)	0.6510(0.0051)
Lasso (`glmnet`)	0.0024(0.0001)	0.0038(0.0001)	0.0065(0.0005)	0.0466(0.0262)

HHSF223201000072C. Lie Wang is supported by NSF Grant DMS-1005539. Xiaoming Yuan is supported by the General Research Fund form Hong Kong Research Grants Council: 203311 and 203712.

References

Belloni, A., Chernozhukov, V. and Wang, L. (2011). Square-root lasso: pivotal recovery of sparse signals via conic programming. *Biometrika* **98** 791–806.

Boyd, S., Parikh, N., Chu, E., Peleato, B. and Eckstein, J. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning* **3** 1–122.

Cai, T., Liu, W. and Luo, X. (2011). A constrained ℓ_1 minimization approach to sparse precision matrix estimation. *Journal of the American Statistical Association* **106** 594–607.

Candes, E. and Tao, T. (2007). The dantzig selector: Statistical estimation when p is much larger than n. *The Annals of Statistics* **35** 2313–2351.

Chen, S. S., Donoho, D. L. and Saunders, M. A. (1998). Atomic decomposition by basis pursuit. *SIAM journal on scientific computing* **20** 33–61.

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

Gabay, D. and Mercier, B. (1976). A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & Mathematics with Applications* **2** 17–40.

He, B. and Yuan, X. (2012a). On non-ergodic convergence rate of douglas-rachford alternating direction method of multipliers. Tech. rep., Nanjing University.

He, B. and Yuan, X. (2012b). On the o(1/n) convergence rate of the douglas-rachford alternating direction method. *SIAM Journal on Numerical Analysis* **50** 700–709.

Liu, H. and Wang, L. (2012). Tiger: A tuning-insensitive approach for optimally estimating gaussian graphical models. Tech. rep., Princeton University.

Liu, J. and Ye, J. (2010). Efficient l_1/l_q norm regularization. Tech. rep., Arizona State University.

Sun, T. and Zhang, C.-H. (2012). Scaled sparse linear regression. *Biometrika* **99** 879–898.

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

Wang, L. (2013). L1 penalized lad estimator for high dimensional linear regression. *Journal of Multivariate Analysis* .