The picasso Package for High Dimensional Regularized Sparse Learning in R

Xingguo Li[*], Jason Ge[*], Mengdi Wang, Tong Zhang, Han Liu, and Tuo Zhao[†]

Abstract

We introduce an R package named picasso, which implements a unified framework of pathwise coordinate optimization for a variety of sparse learning problems (Sparse Linear Regression, Sparse Logistic Regression and Sparse Poisson Regression), combined with efficient active set selection strategies. Besides, the package allows users to choose different sparsity-inducing regularizers, including the convex ℓ_1, nonvoncex MCP and SCAD regularizers. The package is coded in C and can scale up to large problems efficiently with the memory optimized using sparse matrix output.

1 Introduction

The pathwise coordinate optimization combined is undoubtedly one the of the most popular solvers for a large variety of sparse learning problems. It takes advantage of the solution sparsity through a simple but elegant algorithmic structure, and significantly boosts the computational performance in practice(Friedman et al., 2007). Some recent advances in (Zhao et al., 2014; Ge et al., 2016) establishes theoretical guarantees to further justify its computational and statistical superiority for both convex and nonvoncex sparse learning, which makes it even more attractive to practitioners.

Here we introduce an R package called picasso, which implements a unified toolkit of pathwise coordinate optimization for a large class of convex and nonconvex regularized sparse learning approaches. Efficient active set selection strategies are provided to guarantee superior statistical and computational preference. Specifically, we implement sparse linear regression, sparse logistic regression, and sparse Poisson regression (Tibshirani, 1996). The options for regularizers include the ℓ_1, MCP, and SCAD regularizers (Fan and Li, 2001; Zhang, 2010). Unlike existing packages implementing heuristic optimization algorithms such as ncvreg, our implemented algorithm picasso have strong theoretical guarantees that it attains a global linear convergence to a unique sparse local optimum with optimal statistical properties (e.g. minimax optimality and oracle properties). See more technical details in Zhao et al. (2014); Ge et al. (2016).

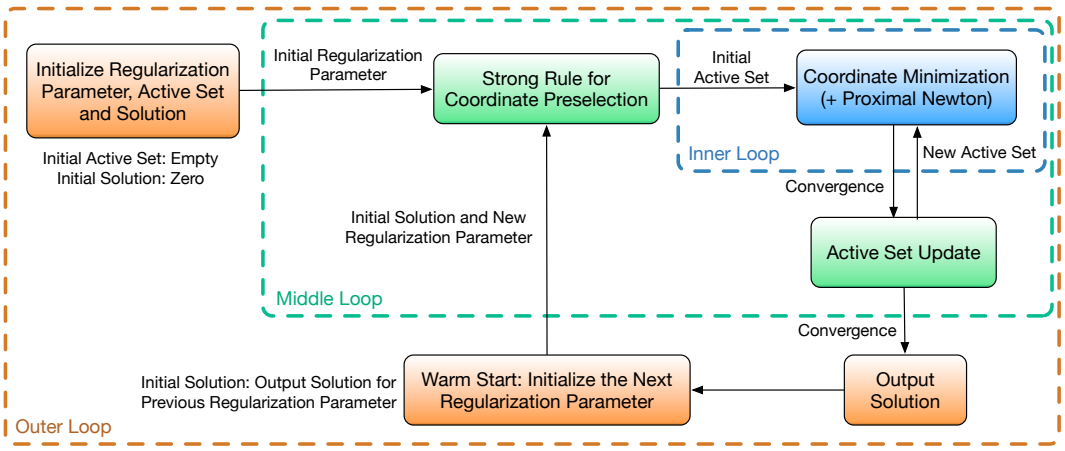


Figure 1: The pathwise coordinate optimization framework with 3 nested loops : (1) Warm start initialization; (2) Active set updating and strong rule for coordinate preselection; (3) Active coordinate minimization.

2 Algorithm Design and Implementation

The algorithm implemented in `picasso` is mostly based on the generic pathwise coordinate optimization framework proposed by Zhao et al. (2014); Ge et al. (2016), which integrates the warm start initialization, active set updating strategy, and strong rule for coordinate preselection into the classical coordinate optimization. The algorithm contains three structurally nested loops as shown in Figure 1:

(1) Outer loop: The warm start initialization, also referred to as the pathwise optimization scheme, is applied to minimize the objective function in a multistage manner using a sequence of decreasing regularization parameters, which yields a sequence of solutions from sparse to dense. At each stage, the algorithm uses the solution from the previous stage as initialization.

(2) Middle loop: The algorithm first divides all coordinates into active ones (active set) and inactive ones (inactive set) by a so-called strong rule based on coordinate gradient thresholding (Tibshirani et al., 2012). Then the algorithm calls an inner loop to optimize the objective, and update the active set based on efficient active set updating strategies. Such a routine is repeated until the active set no longer changes

(3) Inner loop: The algorithm conducts coordinate optimization (for sparse linear regression) or proximal Newton optimization combined with coordinate optimization (for sparse logistic regression and Possion regression) only over active coordinates until convergence, with all inactive coordinates staying zero values. The active coordinates are updated efficiently using an efficient "naive update" rule that only operates on the non-zero coefficients. Further efficiencies are achieved using the "covariance update" rule. See more details in (Friedman

*Xingguo Li and Jason Ge contributed equally.
†Corresponding Author

et al., 2010). The inner loop terminates when the successive descent is within a predefined numerical precision.

The warm start initialization, active set updating strategies, and strong rule for coordinate preselection significantly boost the computational performance, making pathwise coordinate optimization one of the most important computational frameworks for sparse learning. The package is implemented in `C` with the memory optimized using sparse matrix output, and called from `R` by a user-friendly interface. The numerical evaluations show that `picasso` is efficient and can scale to large problems.

3 Examples of User Interface

We illustrate the user interface by analyzing the eye disease data set in `picasso`.

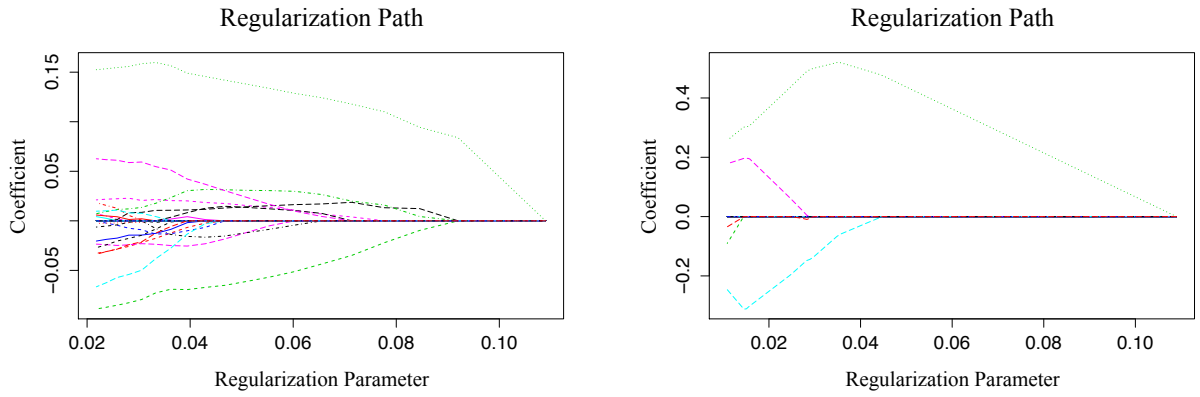
```
> # Load the data set
> library(picasso); data(eyedata)
> # Lasso
> out1 = picasso(x,y,method="l1",opt="naive",nlambda=20,
+                lambda.min.ratio=0.2,max.act.in=3)
> # MCP regularization
> out2 = picasso(x,y,method="mcp",opt="cov",df=100,max.act.in=3)
> # Plot solution paths
> plot(out1); plot(out2)
```

The program automatically generates a sequence of regularization parameters and estimate the corresponding solution paths based on the ℓ_1 and MCP regularizers respectively. For the ℓ_1 regularizer, the number of regularization parameters as 20, and the minimal regularization parameter as `0.2*lambda.max`. Here `lambda.max` is the smallest regularization parameter yielding an all zero solution (automatically calculated the package). For the MCP regularizer, we choose the "covariance update" for the sparse update in the inner loop and the maximal degree of freedom (nonzero coefficients in the solution) to be 100. Here `nlambda` and `lambda.min.ratio` are omitted, and therefore set by the default values (`nlambda=100` and `lambda.min.ratio=0.01`). We further plot two solution paths in Figure 3.

4 Numerical Simulation

To demonstrate the superior efficiency of our package, we compare `picasso` with `R` package `ncvreg`, which is a popular existing package for nonconvex regularized sparse regression. All experiments are evaluated on a PC with Intel Core i5 3.2GHz processor. Timings of the CPU execution are recored in seconds and averaged over 100 replications on a sequence of 50 regularization parameters with approximately the same estimation errors. The convergence threshold are chosen to be 10^{-7} for all experiments.

We first compare the timing performance and the statistical performance for sparse linear regression under well-conditioned scenarios. We choose the (n, d) pairs as $(500, 5000)$ and $(1000, 10000)$ respectively, where n is the number of observation in the response vector $y \in \mathbb{R}^n$ and d is the dimension of the parameter vector $\theta \in \mathbb{R}^d$. We also set `opt="naive"`. For the design matrix $X \in \mathbb{R}^{n \times d}$,



(a) ℓ_1 (b) MCP

we generate each row independently from a d-dimensional normal distribution $\mathcal{N}(\mathbf{0}, \Sigma)$, where $\Sigma_{ij} = 0.5$ for $i \neq j$ and $\Sigma_{ii} = 1$. Then we have $y = X\theta + \varepsilon$, where θ has all 0 entries except $\theta_{150} = 2$, $\theta_{380} = 3$, $\theta_{690} = -1.5$ and $\varepsilon \in \mathbb{R}^n$ has independent $\mathcal{N}(0, 1)$ entries. From the summary in Table 1, we see that while achieving almost identical optimal estimation errors $\|\theta - \widehat{\theta}\|_2$, `picasso` uniformly outperforms `ncvreg` under all settings, where `picasso` is approximately $50 \sim 100$ times faster.

We then compare the timing performance and the statistical performance for sparse linear regression under ill-conditioned scenarios. We choose the (n, d) pairs as $(50, 5000)$ and $(50, 10000)$ respectively. The generations of X, θ and ε are identical to the settings above, except that $\Sigma_{ij} = 0.75$ for $i \neq j$. Due to the choices that values of d are much larger than n, and a larger value is chosen for Σ_{ij} for $i \neq j$, the problems considered here are much more challenging than the problems in the well-conditioned scenarios. We see from Table 1 that though `picasso` is slightly slower than `ncvreg`, its statistical performance is much better than `ncvreg`.

We also compare the timing performance for sparse logistic regression. The choices of (n, d) pairs are $(500, 2000)$, $(1000, 2000)$, $(500, 5000)$ and $(1000, 5000)$. The generations of X and θ follow from the settings for sparse linear regression under well-conditioned scenarios. Then the response vector y has independent Bernoulli$\left(\frac{\exp(X_{i*}^\top \theta)}{1+\exp(X_{i*}^\top \theta)}\right)$ entries. We see from Table 2 that `picasso` outperforms `ncvreg` under all settings, and scales better for increasing values of n and d.

We want to make a final comment that further speedups may be achieved for sparse linear regression with less correlated settings of the design matrix. For example, when the rows of X are generated independently from some multivariate normal distribution with $\Sigma_{ij} = a^{|i-j|}$ for some constant $a \in (0, 1)$, then we may achieve > 100 times of acceleration than `ncvreg` by setting `opt="cov"` and `df` to be small compared with $\min\{n, d\}$.

5 Conclusion

The `picasso` package demonstrates significantly improved computational and statistical performance over existing packages such as `ncvreg` for nonconvex regularized sparse learning. Moreover, the algorithm implemented in `picasso`, which guarantees a global linear convergence to a unique sparse local optimum with optimal statistical properties. Overall, the `picasso` package has the potential to serve as a powerful toolbox for high dimensional nonconvex sparse learning.

Table 1: Average timing performance (in seconds) and optimal estimation errors with standard errors in the parentheses on sparse linear regression.

Sparse Linear Regression (Well-Conditioned)					
Method	Package	$n = 500, d = 5000$		$n = 1000, d = 10000$	
		Time	Est. Err.	Time	Est. Err.
ℓ_1 norm	picasso	0.5013(0.1404)	0.3924(0.0662)	1.4040(0.2358)	0.2677(0.0346)
	ncvreg	42.521(7.7725)	0.3924(0.0667)	138.44(24.122)	0.2670(0.0345)
MCP	picasso	0.4957(0.1809)	0.0773(0.0499)	1.3815(0.2018)	0.0586(0.0306)
	ncvreg	22.290(2.7846)	0.0775(0.0499)	94.746(18.329)	0.0592(0.0308)
SCAD	picasso	0.4942(0.0875)	0.0766(0.0505)	1.4384(0.1883)	0.0587(0.0306)
	ncvreg	38.476(7.0584)	0.0769(0.0505)	139.59(25.226)	0.0591(0.0309)
Sparse Linear Regression (Ill-Conditioned)					
Method	Package	$n = 50, d = 5000$		$n = 50, d = 10000$	
		Time	Est. Err.	Time	Est. Err.
MCP	picasso	0.1480(0.0098)	0.4629(0.2840)	0.2181(0.0310)	0.4904(0.3232)
	ncvreg	0.0908(0.0053)	1.5069(0.9596)	0.1646(0.0087)	1.7827(0.8856)

Table 2: Average timing performance (in seconds) with standard errors in the parentheses on sparse logistic regression.

Sparse Logistic Regression					
Method	Package	$d = 2000$		$d = 5000$	
		$n = 500$	$n = 1000$	$n = 500$	$n = 1000$
ℓ_1 norm	picasso	0.2127(0.0089)	0.3918(0.0252)	0.4583(0.0321)	0.8054(0.0246)
	ncvreg	1.2464(0.7255)	5.7377(1.5040)	2.2527(0.7114)	10.096(2.4513)
MCP	picasso	0.3820(0.0892)	0.4860(0.0282)	0.6197(0.0543)	0.9942(0.0710)
	ncvreg	0.6639(0.4253)	2.5244(0.9032)	0.8451(0.2590)	2.8319(0.8218)
SCAD	picasso	0.3383(0.0553)	0.4995(0.0575)	0.6188(0.0555)	0.9323(0.0711)
	ncvreg	0.7226(0.1639)	3.9026(0.9745)	1.5180(0.5561)	7.1200(0.8744)

We will continue to maintain and support this package.

References

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

Friedman, J., Hastie, T., Höfling, H. and Tibshirani, R. (2007). Pathwise coordinate optimization. *The Annals of Applied Statistics* **1** 302–332.

Friedman, J., Hastie, T. and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software* **33** 1–13.

Ge, J., Wang, M., Liu, H., Hong, M. and Zhao, T. (2016). Homotopy active set proximal newton algorithm for sparse learning. Tech. rep., Georgia Tech.

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

Tibshirani, R., Bien, J., Friedman, J., Hastie, T., Simon, N., Taylor, J. and Tibshirani, R. (2012). Strong rules for discarding predictors in lasso-type problems. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **74** 245–266.

Zhang, C. (2010). Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics* **38** 894–942.

Zhao, T., Liu, H. and Zhang, T. (2014). Pathwise coordinate optimization for nonconvex sparse learning: Algorithm and theory. *arXiv preprint arXiv:1412.7477* .