Fast Algorithms for LAD Lasso Problems

Robert J. Vanderbei

2015 Nov 3

INFORMS
Philadelphia

http://www.princeton.edu/∼rvdb

Lasso Regression

The problem is to solve a sparsity-encouraging "regularized" regression problem:

$$\text{minimize} \quad \|Ax - b\|_2^2 + \lambda \|x\|_1$$

My gut reaction:

Replace *least squares* (LS) with *least absolute deviations* (LAD).

LAD is to LS as median is to mean. Median is a more *robust* statistic.

The LAD version can be recast as a *linear programming* (LP) problem.

If the solution is expected to be sparse, then the *simplex method* can be expected to solve the problem very quickly.

No one knows the "correct" value of the parameter λ. The *parametric simplex method* can solve the problem for *all values of* λ from $\lambda = \infty$ to a small value of λ in the same (fast) time it takes the standard simplex method to solve the problem for *one choice* of λ.

The parametric simplex method can be stopped when the desired sparsity level is attained. No need to guess/search-for the correct value of λ.

Linear Programming Formulation

The *least absolute deviations* variant is given by

$$\text{minimize} \quad \|Ax - b\|_1 + \lambda \|x\|_1$$

It is easy to reformulate this problem as a linear programming problem:

$$\min_x \; 1^T \varepsilon + \lambda \, 1^T \delta$$

$$\begin{aligned} \text{subject to} \quad -\varepsilon &\leq Ax - b \leq \varepsilon \\ -\delta &\leq x \leq \delta \end{aligned}$$

Note: There exists a critical value λ_c such that the optimal solution for all $\lambda \geq \lambda_c$ is trivial:

$$\begin{aligned} x &= 0 \\ \delta &= 0 \\ \varepsilon_i &= |b_i|, \qquad \text{for all } i. \end{aligned}$$

The Simplex Method

We start by introducing slack variables:

$$
\begin{array}{lrcrcrcrcrcl}
\text{minimize} & & & 1^T\varepsilon & + & \lambda\, 1^T\delta & & & & & & \\
\text{subject to} & -Ax & - & \varepsilon & & & + & s & & & = & -b \\
& Ax & - & \varepsilon & & & & & + & t & = & b \\
& -x & & & - & \delta & & & + & u & = & 0 \\
& x & & & - & \delta & & & & & + v = & 0 \\
& & & & & & & & & & s,\ t,\ u,\ v \geq & 0
\end{array}
$$

We must identify a partition of the set of all variables into *basic* and *nonbasic* variables.

If A is an $m \times n$ matrix, then there are $2m + 2n$ equality constraints in $3m + 4n$ variables.

The variables, x, ε, and δ can be regarded as free variables.

In the simplex method, free variables are always basic variables.

Let $\mathcal{P}$ denote the set of indices i for which $b_i \geq 0$ and let $\mathcal{N}$ denote those for which $b_i < 0$.

Dictionary Form

Writing the equations with the nonbasic variables defining the basic ones, we get:

minimize			$1^T b_{\mathcal{P}} - 1^T b_{\mathcal{N}}$	$+$	$1^T s_{\mathcal{P}}$	$+$	$1^T t_{\mathcal{N}}$	$-$	$\frac{1}{2}(p-n)u$	$+$	$\frac{1}{2}(p-n)v$
								$+$	$\lambda\ \frac{1}{2}u$	$+$	$\lambda\ \frac{1}{2}v$
subject to	$\varepsilon_{\mathcal{P}}$	$=$	$b_{\mathcal{P}}$	$+$	$s_{\mathcal{P}}$			$-$	$\frac{1}{2}Pu$	$+$	$\frac{1}{2}Pv$
	$s_{\mathcal{N}}$	$=$	$-2b_{\mathcal{N}}$			$+$	$t_{\mathcal{N}}$	$+$	Nu	$-$	Nv
	$t_{\mathcal{P}}$	$=$	$2b_{\mathcal{P}}$	$+$	$s_{\mathcal{P}}$			$-$	Pu	$+$	Pv
	$\varepsilon_{\mathcal{N}}$	$=$	$-b_{\mathcal{N}}$			$+$	$t_{\mathcal{N}}$	$+$	$\frac{1}{2}Nu$	$-$	$\frac{1}{2}Nv$
	x	$=$	0					$+$	$\frac{1}{2}u$	$-$	$\frac{1}{2}v$
	δ	$=$	0					$+$	$\frac{1}{2}u$	$+$	$\frac{1}{2}v$

where

$$A = \begin{bmatrix} P \\ N \end{bmatrix},\ \varepsilon = \begin{bmatrix} \varepsilon_{\mathcal{P}} \\ \varepsilon_{\mathcal{N}} \end{bmatrix},\ s = \begin{bmatrix} s_{\mathcal{P}} \\ s_{\mathcal{N}} \end{bmatrix},\ t = \begin{bmatrix} t_{\mathcal{P}} \\ t_{\mathcal{N}} \end{bmatrix},\ p = 1^T P, \text{ and } n = 1^T N.$$

Dictionary Form

Writing the equations with the nonbasic variables defining the basic ones, we get:

minimize		$1^T b_{\mathcal{P}} - 1^T b_{\mathcal{N}}$	$+$	$1^T s_{\mathcal{P}}$	$+$	$1^T t_{\mathcal{N}}$	$-$	$\frac{1}{2}(p-n)u$	$+$	$\frac{1}{2}(p-n)v$
							$+$	$\lambda\ \frac{1}{2}u$	$+$	$\lambda\ \frac{1}{2}v$
subject to										
	$s_{\mathcal{N}}$	$=$ $-2b_{\mathcal{N}}$			$+$	$t_{\mathcal{N}}$	$+$	Nu	$-$	Nv
	$t_{\mathcal{P}}$	$=$ $2b_{\mathcal{P}}$	$+$	$s_{\mathcal{P}}$			$-$	Pu	$+$	Pv

where

$$A = \begin{bmatrix} P \\ N \end{bmatrix},\ \varepsilon = \begin{bmatrix} \varepsilon_{\mathcal{P}} \\ \varepsilon_{\mathcal{N}} \end{bmatrix},\ s = \begin{bmatrix} s_{\mathcal{P}} \\ s_{\mathcal{N}} \end{bmatrix},\ t = \begin{bmatrix} t_{\mathcal{P}} \\ t_{\mathcal{N}} \end{bmatrix},\ p = 1^T P, \text{ and } n = 1^T N.$$

Parametric Simplex Method

A *dictionary solution* is obtained by setting the *nonbasic* variables to zero and reading off the values of the *basic* variables from the dictionary.

The dictionary solution on the previous slide is *feasible*.

It is optimal provided the coefficients of the variables in the objective function are nonnegative:

$$\begin{aligned} 1 &\geq 0, \\ -(p_j - n_j)/2 + \lambda/2 &\geq 0, \\ (p_j - n_j)/2 + \lambda/2 &\geq 0. \end{aligned}$$

This simplifies to

$$\lambda \geq \max_j (p_j - n_j)$$

The specific index defining this max sets the lower bound on λ and identifies the *entering variable* for the parametric simplex method.

The *leaving variable* is determined in the usual way.

A simplex pivot is performed.

A new range of λ values is determined (the lower bound from before becomes the upper bound).

The process is repeated.

Random Example

A matrix A with $m = 3000$ observations and $n = 200$ was generated randomly.

The linear programming problem has 6400 constraints, 9800 variables, and 1213200 nonzeros in the constraint matrix.

A simple C implementation of the algorithm finds a solution with 6 nonzero regression coefficients in just 44 simplex pivots (7.8 seconds).

This compares favorably with the 5321 pivots (168.5 seconds) to solve the problem all the way to $\lambda = 0$.

A speedup by a factor of 21.6.

"Real-World" Example

A regression model from my *Local Warming* studies with $m = 2899$ observations and $n = 106$ regression coefficients of which only 4 are deemed relevant (linear trend plus seasonal changes).

The linear programming problem has 6010 constraints, 9121 variables, and 626820 nonzeros in the constraint matrix.

My implementation finds a solution with 4 nonzero regression coefficients in 1871 iterations (37.0 seconds).

This compares favorably with the 4774 iterations (108.2 seconds) to solve the problem to $\lambda = 0$.

But, why so many iterations just to get just a few nonzeros?

A Simple Median Example

To answer the question, consider the simplest example where $n = 1$ and $A = e$.

In this case, the problem is simply a lasso variant of a median computation...

$$\min_x \sum_i |x - b_i| + \lambda|x|.$$

Clearly, if λ is a positive *integer*, then this Lasso problem is equivalent to computing the median of the original numbers, $b_1, b_2, \ldots, b_m$, augmented with λ zeros, $0, 0, 0, \ldots, 0$.

Best case scenario: the median of the b_i's is close to zero. For example, suppose the median is positive but the next smaller b_i is negative. Then, just need to add one or two zeros to make the median exactly zero. And, after one simplex iteration, we will arrive at the true median.

Worst case scenario: all of the b_i's have the same sign. Suppose they are all positive. Then we have to add $m+1$ zeros to make the lasso'ed median zero. The first iteration will produce a single nonzero: *the minimum of the b_i's*. That's a *terrible* estimate of the median! Each subsequent simplex pivot produces the next larger b_i until, after $m/2$ pivots, the true median is found.

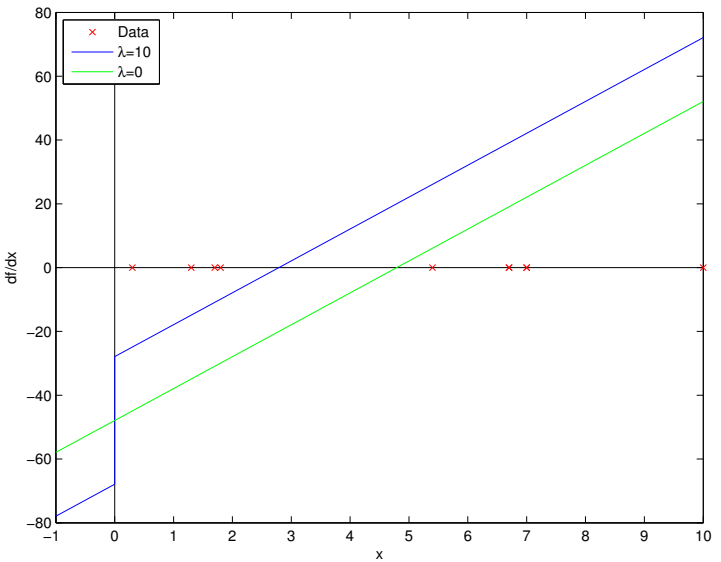
Traditional (Least Squares) Lasso

Consider the least squares variant of the previous example:

$$\min_x \ \frac{1}{2}\sum_i (x - b_i)^2 + \lambda |x|.$$

Set the derivative of the objective function to zero:

$$f'(x) = mx - \sum_i b_i + \lambda \ \mathrm{sgn}(x) = 0.$$



Without loss of generality, suppose that $\sum_i b_i > 0$.
We see that the Lasso solution is zero for

$$\lambda \geq \sum_i b_i.$$

For $0 \leq \lambda \leq \sum_i b_i$, the solution is

$$x = \frac{\sum_i b_i - \lambda}{m}.$$

So, as with LAD-Lasso, depending on the choice of λ we can get anything between 0 and the *sample mean* $\bar{x} = \sum_i b_i / m$.

(Unexpected) Conclusion

The *parametric simplex method* for the *LAD-Lasso* problem can generate sparse solutions in a very small number of pivots.

With the parametric simplex method there is no need to guess/search for the value of λ that gives a desired regressant sparsity.

When the number of pivots is small, one can expect the result to be good.

But, LAD-Lasso regression can also sometimes produce *terrible* results.

The same is true for traditional Lasso regression.

Lasso References

- *Regression Shrinkage and Selection via the LASSO*,
  Tsibshirani,
  J. Royal Statistical Soc., Ser. B, 58, 267–288, 1996

- *Robust Regression Shrinkage and Consistent Variable Selection Through the LAD-Lasso*,
  Wang, Li, and Jiang,
  Journal of Business and Economic Statistics, 25(3), 347–355, 2007.

Parametric Simplex Method References

- *Linear Programming and Extensions*,
  Dantzig
  Princeton University Press, Chapter 11, 1963

- *Linear Programming: Foundations and Extensions*,
  Vanderbei, any edition
  Springer, Chapter 7, 1997

- *Frontiers of Stochastically Nondominated Portfolios*,
  Ruszczyński and Vanderbei,
  Econometrica, 71(4), 1289–1297, 2003.

- *The Fastclime Package for Linear Programming and Large-Scale Precision Matrix Estimation in R*,
  Pang, Liu, and Vanderbei,
  Journal of Machine Learning Research, 2013.

- *Optimization for Compressed Sensing: the Simplex Method and Kronecker Sparsification*,
  Vanderbei, Liu, Wang, and Lin,
  Mathematical Programming Computation, to appear (hopefully), 2016.

- *A Parametric Simplex Approach to Statistical Learning Problems*,
  Pang, Zhao, Vanderbei, Liu,
  in preparation, 2015.

Thank You!