Elastic Net Minimization as Non-Negative Least Squares using the Landweber Iteration

Andrew E. Yagle
Department of EECS, The University of Michigan, Ann Arbor, MI 48109-2122

Abstract— **Minimization of the elastic net cost function $||y-Ax||_2^2+2\lambda||x||_1+2\mu||x||_2^2$ arises in reconstruction of sparse signals from noisy observations y of underdetermined linear combinations Ax of x. We reformulate this problem as a non-negative least-squares problem, and solve the latter using Landweber iteration with a non-negativity constraint. In particular, this yields a simple derivation of the thresholded Landweber iteration for minimization of the LASSO cost function.**

Keywords— **Sparse reconstruction**
Phone: 734-763-9810. Fax: 734-763-1503.
Email: aey@eecs.umich.edu. **EDICS: 2-REST.**

I. INTRODUCTION

A. Problem Statement

We are given the noisy observations

$$y = Ax + w \tag{1}$$

- x is an unknown sparse (mostly zero) N-vector;
- A is a known $M \times N$ full-rank matrix, where
- $M << N$ so the problem is underdetermined;
- w is a M-vector of zero-mean uncorrelated Gaussian (noise) random variables with unit variances.
- Each component x_i of x is a random variable with

$$f_{x_i}(X) = C\,\underbrace{e^{-\lambda|X|}}_{\text{SPARSE}}\,\underbrace{e^{-\mu X^2}}_{\text{SMALL}}. \tag{2}$$

The Gaussian prior penalizes too-large values of x. The Laplacian (two-sided exponential) sparsifies x. The negative log-likehood function for (1) is then the *elastic net* function (H. Zou and T. Hastie, 2005)

$$L = \underbrace{(1/2)||y-Ax||_2^2}_{\text{NOISE}} + \underbrace{\lambda||x||_1}_{\text{LASSO}} + \underbrace{\mu||x||_2^2}_{\text{RIDGE}} \tag{3}$$

where we have defined the usual two norms

$$||x||_1 = \sum_{i=1}^{N} |x_i| \quad \text{and} \quad ||x||_2^2 = \sum_{i=1}^{N} x_i^2. \tag{4}$$

The elastic net has some advantages over LASSO. It can identify multiple nonzero x_i corresponding to closely correlated columns of A, unlike LASSO.

The elastic net function includes as special cases

λ	μ	CRITERION
$=0$	$=0$	Least squares
$=0$	$\neq 0$	Tikhonov
$\to 0$	$=0$	Basis pursuit
$\neq 0$	$=0$	LASSO
$\neq 0$	$\neq 0$	Elastic net

Tikhonov, also known as ridge, provides regularization if A is near-singular and noise levels are high.

Basis pursuit minimizes $||x||_1$ subject to the constraint $y{=}Ax$, and produces a sparse solution x.

LASSO (Least Absolute Shrinkage and Selection Operator) produces a sparse solution x in noise.

The ℓ_1-norm penalty term $||x||_1$ penalizes small deviations of the elements of x from zero. A considerable amount of research since 2000 has proven what the geophysical community has observed since the 1960s: The ℓ_1 norm produces sparse solutions. The ℓ_1 norm produces the sparsest solution if the number of nonzero elements of x is sufficiently small.

The ℓ_2-norm penalty term $||x||_2^2$ does not penalize small deviations, since its slope is zero at zero. But it does penalize large deviations more heavily than the ℓ_1 norm, and thus stabilizes the solution in the presence of noise. Use of an ℓ_2 norm penalty term for this purpose is called Tikhonov regularization.

The minimum elastic net solution has been computed using coordinate descent, in which all but one variable x_i is held constant, and the x_i minimizing the elastic net function L is computed in closed form. Since the elastic net functional is convex and is the sum of a differentiable part and a separable non-differentiable part $\lambda||x||_1$, coordinate descent is guaranteed to converge to the minimizer of L.

However, in many applications A is not represented as a matrix, but as a sequence of operations, such as wavelet or fast Fourier transforms. In this case, Ax and $A^T y$ can be computed much more quickly than a typical matrix-vector multiplication, e.g., $N \log N$. This motivates use of the Landweber iteration below.

II. Reformulation of Elastic Net Minimization as Non-Negative Least Squares

First, we use the usual procedure of defining the positive x^+ and negative x^- parts of x as

$$x_i^+ = \begin{cases} +x_i & \text{if } x_i \geq 0 \\ 0 & \text{if } x_i \leq 0 \end{cases} \quad x_i^- = \begin{cases} -x_i & \text{if } x_i \leq 0 \\ 0 & \text{if } x_i \geq 0 \end{cases} \geq 0. \tag{5}$$

Then we have

$$\begin{aligned} x &= x^+ - x^- \quad (6) \\ ||x||_1 &= \sum_{i=1}^{N} (x_i^+ + x_i^-) \\ ||x||_2^2 &= ||x^+||_2^2 + ||x^-||_2^2 \end{aligned}$$

Now consider the still-underdetermined problem

$$\underbrace{\begin{bmatrix} y \\ -\frac{\lambda}{\sqrt{2\mu}}\mathbf{1} \end{bmatrix}}_{\tilde{y}} = \underbrace{\begin{bmatrix} A & -A \\ \sqrt{2\mu}I & \sqrt{2\mu}I \end{bmatrix}}_{\tilde{A}} \underbrace{\begin{bmatrix} x^+ \\ x^- \end{bmatrix}}_{\tilde{x}} \tag{7}$$

where $\mathbf{1}$=[1,1…1]^T, and both $x_i^+ \geq 0$ and $x_i^- \geq 0$.

The squared ℓ_2 error is then (note that $x_i^+ x_i^-$=0)

$$\begin{aligned} ||\tilde{y} - \tilde{A}\tilde{x}||_2^2 &= ||y - A(x^+ - x^-)||_2^2 \quad (8) \\ &+ ||\sqrt{2\mu}(x^+ + x^-) + \frac{\lambda}{\sqrt{2\mu}}||_2^2 \\ &= ||y - Ax||_2^2 \\ &+ 2\mu||x||_2^2 + 2\lambda||x||_1 + \frac{N\lambda^2}{2\mu} \\ &= 2L + \frac{N\lambda^2}{2\mu} \end{aligned}$$

The final term in (8) does not affect the argmax, so computing the *non-negative least-squares* solution to (7) minimizes the elastic net cost function L.

III. Solution of the Non-Negative Least-Squares Using Landweber

A. Review of Landweber Iteration

The basic Landweber iteration is

$$x^{k+1} = x^k + A^T(y - Ax), \quad x^0 = 0 \tag{9}$$

where x^k is the estimate of x at the k^{th} iteration. The Landweber iteration can be viewed as a steepest descent algorithm for minimizing the cost function

$$\begin{aligned} f(x) &= (1/2)||y - Ax||_2^2 \\ \nabla f(x) &= -A^T(y - Ax). \end{aligned} \tag{10}$$

The basic steepest descent algorithm is

$$\begin{aligned} x^{k+1} &= x^k - \nabla f(x) \\ &= x^k + A^T(y - Ax) \end{aligned} \tag{11}$$

which is the basic Landweber iteration.

Another way of looking at this is to note that $\nabla f(x)$ is the correlation of the residual Ax–y with each column of A. The bigger this correlation, the more we should alter that component of x^k.

B. Non-Negative Landweber Iteration

Here, we use the basic Landweber iteration

$$\tilde{x}^{k+1} = \tilde{x}^k + \tilde{A}^T(\tilde{y} - \tilde{A}\tilde{x}^k) \tag{12}$$

with a non-negativity constraint at each iteration

$$\tilde{x}_i^{k+1} = \max[\tilde{x}_i^{k+1}, 0]. \tag{13}$$

We call this the *non-negative* Landweber iteration.

Since the cost functional $f(x)$ and the non-negativity constraints $x_i^+ \geq 0$ and $x_i^- \geq 0$ are all convex, the non-negative Landweber iteration is guaranteed to converge if the maximum eigenvalue of $\tilde{A}^T\tilde{A} < 2$. The nonzero eigenvalues of $\tilde{A}^T\tilde{A}$ are the eigenvalues of $\tilde{A}\tilde{A}^T$, which from

$$\begin{aligned} \tilde{A}\tilde{A}^T &= \begin{bmatrix} A & -A \\ \sqrt{2\mu}I & \sqrt{2\mu}I \end{bmatrix} \begin{bmatrix} A^T & \sqrt{2\mu}I \\ -A^T & \sqrt{2\mu}I \end{bmatrix} \\ &= \begin{bmatrix} 2AA^T & 0 \\ 0 & 4\mu I \end{bmatrix} \end{aligned} \tag{14}$$

are double the eigenvalues of AA^T, and 4μ. The non-negative Landweber iteration will converge if $\mu < \frac{1}{2}$ and all the singular values of A are less than unity.

Substitution of (7) in (12) gives

$$\begin{aligned} z^{k+1} &= y - A[(x^+)^k - (x^-)^k] \\ v^{k+1} &= \lambda\mathbf{1} + 2\mu[(x^+)^k + (x^-)^k] \\ (x^+)^{k+1} &= (x^+)^k + A^T z^{k+1} - v^{k+1} \\ (x^-)^{k+1} &= (x^-)^k - A^T z^{k+1} - v^{k+1} \end{aligned} \tag{15}$$

followed by the non-negativity constraints

$$\begin{aligned} (x^+)_i^{k+1} &= \max[(x^+)_i^{k+1}, 0] \\ (x^-)_i^{k+1} &= \max[(x^-)_i^{k+1}, 0]. \end{aligned} \tag{16}$$

so most of the computation in the Landweber iteration are the matrix-vector multiplications Ax and $A^T z$. These can often be implemented using a fast algorithm such as the fast Fourier transform, the fast wavelet algorithm or a sparse matrix-times-vector.

C. Derivation of Thresholded Landweber for LASSO

Let μ=0 in the elastic net criterion (3). This gives the LASSO criterion. We now examine what this does to the non-negative Landweber iteration (15).

Let μ=0 in (15). This gives the iteration

$$\begin{aligned} z^{k+1} &= y - A[(x^+)^k - (x^-)^k] \\ (x^+)^{k+1} &= (x^+)^k + A^T z^{k+1} - \lambda\mathbf{1} \\ (x^-)^{k+1} &= (x^-)^k - A^T z^{k+1} - \lambda\mathbf{1} \end{aligned} \tag{17}$$

followed by the non-negativity constraints

$$\begin{aligned} (x^+)_i^{k+1} &= \max[(x^+)_i^{k+1}, 0] \\ (x^-)_i^{k+1} &= \max[(x^-)_i^{k+1}, 0]. \end{aligned} \tag{18}$$

Since x^k=$(x^+)^k$–$(x^-)^k$, and $-(x^-)^k$ is the negative values of x^k, (17) is the usual Landweber iteration applied to (1), followed by *shrinkage* of $|x^k|$ by λ and *thresholding* values of $|x^k| < \lambda$ to 0. This is the well-known *thresholded Landweber iteration*

$$x^{k+1} = x^k + A^T(y - Ax), \quad x^0 = 0 \tag{19}$$

followed by shrinkage and thresholding

$$x_i^{k+1} = \begin{cases} x_i^{k+1} - \lambda & \text{if } x_i^{k+1} > +\lambda \\ x_i^{k+1} + \lambda & \text{if } x_i^{k+1} < -\lambda \\ 0 & \text{if } |x_i^{k+1}| < \lambda. \end{cases} \tag{20}$$

This is a much simpler derivation of thresholded Landweber iteration than the usual derivation.

D. Overdetermined A with Orthonormal Columns

Now let (1) no longer be underdetermined, so that $M \geq N$. Let $\sqrt{2}A$ have orthonormal columns, so that

$$A^T A = (1/2)I; \quad \mu = 1/4. \tag{21}$$

Then the non-negative Landweber iteration becomes

$$\begin{aligned} \tilde{x}^{k+1} &= \tilde{x}^k + \tilde{A}^T(\tilde{y} - \tilde{A}\tilde{x}^k) \\ &= \tilde{A}^T\tilde{y} + (I - \tilde{A}^T\tilde{A})\tilde{x}^k \\ &= \tilde{A}^T\tilde{y} = \begin{bmatrix} A^T y - \lambda\mathbf{1} \\ -A^T y - \lambda\mathbf{1} \end{bmatrix} \end{aligned} \tag{22}$$

since in this case $\tilde{A}^T\tilde{A}$=I from

$$\begin{aligned} \tilde{A}^T\tilde{A} &= \begin{bmatrix} A^T & \sqrt{2\mu}I \\ -A^T & \sqrt{2\mu}I \end{bmatrix} \begin{bmatrix} A & -A \\ \sqrt{2\mu}I & \sqrt{2\mu}I \end{bmatrix} \qquad (23) \\ &= \begin{bmatrix} A^T A + 2\mu I & 2\mu I - A^T A \\ 2\mu I - A^T A & A^T A + 2\mu I \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}. \end{aligned}$$

Imposing non-negativity gives the final answer as

$$x_i = \begin{cases} (A^T y)_i - \lambda & \text{if } (A^T y)_i > +\lambda; \\ (A^T y)_i + \lambda & \text{if } (A^T y)_i < -\lambda; \\ 0 & \text{if } |(A^T y)_i| < \lambda. \end{cases} \tag{24}$$

That is, compute the least-squares solution in closed form, and then apply shrinkage and thresholding. Of course, this is also a well-known result for μ=0. Here we have extended it to a nonzero value of μ, which can be used as initialization for other values of μ.

IV. Solution of the Non-Negative Least-Squares Problem Using an Active-Sets Algorithm

A. Active-Sets Algorithm When Solution is Sparse

Of course, the non-negative least-squares problem (7) can also be solved using active-set algorithms, such as the original algorithm of Lawson and Hansen. The idea behind these algorithms is that non-zero (strictly positive) elements of the solution are recursively identified from the largest value of the dual

$$w = A^T(y - A\hat{x}) \tag{25}$$

where $\hat{x}$ is the least-squares solution using the non-zero elements identified at that recursion.

This algorithm requires computation of a least-squares solution for a reduced set of identified non-zreo variables at each recursion. However, in the present problem, *most elements of* $\tilde{x}$ *are known to be 0*, since minimization of L with $\lambda \neq 0$ sparsifies the solution $\tilde{x}$, whose sparsity is the same as the sparsity of x. In fact, the sizes of the least-squares problems to be solved at each recursion do not exceed M.

B. Tiny Numerical Example

The following Matlab code implements the new algorithm. For random matrices with zero-mean Gaussian entries, it seems to works about half the time. For larger problems, the tolerance for `lsnonneg` usually must be raised to foster faster convergence.

```
clear;M=10;N=20;L=.01;E=.0000005;
X(3)=1;X(7)=-2;X(13)=3;X(17)=-4;
e=sqrt(2*E);randn('state',2);
A=randn(M,N);X(N)=0;X=X';Y=A*X;
%GOAL: Compute sparse X from Y.
%NOTE: Doesn't work for all A.
AA=[A -A;e*eye(N) e*eye(N)];
YY=[Y;-L/e*ones(N,1)];
Z=lsnonneg(AA,YY);
[X Z(1:N)-Z(N+1:2*N)]
```

C. Comparison with Non-Negative Sparse x

Since 2008, it has been shown that a sufficiently sparse *and non-negative* x can be computed by solving the non-negative linear problem y=$Ax, x \geq 0$. In practice, this would likely be solved as a non-negative least-squares problem. In the present paper, we have shown that we can use the same algorithm to solve for sparse x, with the following additional benefits:

- Sparse solution x can now have mixed signs;
- ℓ_1-norm minimization to promote sparsity;
- LASSO penalty to deak with noise in the data;
- Ridge penalty to improve conditioning of AA^T.