Subgradient Methods Applied to LASSO Regression

Kasey Tian
Electrical and Computer Engineering
Rutgers University
New Brunswick, NJ, USA
kasey.tian@rutgers.edu

Abstract

Subgradient methods

Index Terms

optimization, subgradients, LASSO, regression

I. Introduction

Subgradient methods are a way to perform an optimization on an objective function which is not fully differentiable. For non-differentiable objective functions, traditional methods, such as gradient descent and Newton's method, are impossible to execute. They can also be combined with a wide variety of other optimization methods, and have far reaching applications. They were originally developed by Shor and others in the Soviet Union in the 1960s and 70s [1] [2]. LASSO regression is a regression analysis method which was first introduced in 1986 [3] in the field of geophysics, but was then independently rediscovered, named, and popularized in 1996 by Tibshirani [4]. It is characterized by regularization using the L_1 norm, which involves the absolute value function and is therefore not fully differentiable. This makes it a prime candidate for applying subgradient methods. To demonstrate a practical implementation LASSO regression will be used with a linear classifier to predict life expectancy from World Health Organization (WHO) data.

II. Mathematical Analysis

In this section we will explore the mathematics of subgradients, subgradient methods and LASSO regression.

A. Defining Subgradient

This section is mostly adapted from [5], with some supplementary material from [1].

1) Definitions: A subgradient is defined for some convex function $f : \mathbb{R}^n \to \mathbb{R}$ at a point $x \in \mathbf{dom}\, f$ as a vector $g \in \mathbb{R}^n$ such that $\forall\, y \in \mathbf{dom}\, f$

$$f(y) \geq f(x) + g^T(y - x) \tag{1}$$

Alternately expressed as

$$f(y) - f(x) \geq g^T(y - x) \tag{2}$$

$$f(x) - f(y) + g^T(y - x) \leq 0 \tag{3}$$

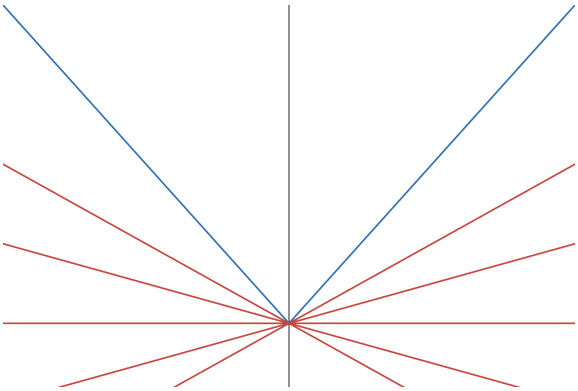


Fig. 1: Absolute value function (blue) with subgradients (red)

There can be multiple subgradients at a point x, so we will also define the subdifferential $\partial f(x)$ to be the set of all subgradients at x.

$$\partial f(x) = \bigcap_{y \in \mathbf{dom}\, f} \left\{ g : f(y) \geq f(x) + g^T(y - x) \right\} \tag{4}$$

If there exists at least one subgradient at a point x, we would say f is subdifferentiable at x. If all points in the domain are subdifferentiable, we say that f is subdifferentiable.

2) Example: Absolute Value: If we consider g in (1) to be a slope, we can visualize a subgradient as being some hyperplane intersecting our function at x for which all values of the function are on or above the plane.

Let us employ this intuition to find a subgradient of the function $f(x) = |x|$ at the point $x = 0$. Graphically, we can see in Fig. 1 that many different lines satisfy this criteron. In fact, we can say that any $g \in [-1, 1]$ would be a subgradient, and therefore $\partial f(0) = [-1, 1]$. But what about other points? For a point $x > 0$, we can surmise that the only possible $g = 1$, as any other value will leave some parts of our function beneath the resulting plane. Likewise for $x < 0$, $g = -1$. Using (4), we can say

$$\partial f(x) = \begin{cases} \{-1\} & x < 0 \\ [-1, 1] & x = 0 \\ \{1\} & x > 0 \end{cases} \tag{5}$$

This can be compared against the derivative of $f(x)$.

$$f'(x) = \begin{cases} -1 & x < 0 \\ 1 & x > 0 \end{cases} \tag{6}$$

We find that where the function is differentiable, the subdifferential contains only the gradient. There is only a difference where the function is not differentiable. Here we find that our set ranges between the two derivatives on either side of it. More formally, we can say that $\partial f(x) = [a, b]$ such that

$$a = \lim_{y \to x^-} \frac{f(y) - f(x)}{y - x} \tag{7}$$

$$b = \lim_{y \to x^+} \frac{f(y) - f(x)}{y - x} \tag{8}$$

for one dimensional functions. We can note that for differentiable points $a = b$, so this definition satisfies our observation up to this point.

3) Properties: There are a few important properties that we shall take note of. Proofs of these properties can be found in Appendix B.

- If and only if the function is differentiable at x, $\partial f(x) = \{\nabla f(x)\}$
- If and only if x^* is a global minimum, $0 \in \partial f(x^*)$
- $\partial f(x)$ will always be a convex set

B. Subgradient Methods

This section is predominantly adapted from [1] and [2].

1) Iteration Step: Subgradient methods are a family of optimization techniques involving the same basic iteration step

$$x^{(k+1)} = x^{(k)} - \alpha_k g^{(k)} \tag{9}$$

where $g^{(k)}$ is any subgradient at the point $x^{(k)}$ and $\alpha_k > 0$. There are a number of different methods of choosing α_k, which will be further explored in Section II-C. This formula can be seen to be very similar to gradient descent. In fact, can be observed that since the only subgradient at differentiable points is the gradient, all subgradient methods *are* gradient descent in these sections, the only difference from regular gradient descent being our way of choosing step size. One notable difference from gradient descent is that subgradient methods are not descent methods; we are not guaranteed that every step will descend. For that reason we should have some method of tracking our best performing point.

$$f_{\text{best}}^{(k)} = \min(f_{best}^{(k-1)}, f(x^{(k)})) \tag{10}$$

And a corresponding $i_{\text{best}}^{(k)}$ such that $f(x^{(i_{\text{best}}^{(k)})}) = f_{\text{best}}^{(k)}$ This would not be needed in a descent method, because our best performing point is always the last point.

2) Convergence Proof: We are focused on converging towards the optimum point within some error of margin. We begin by making the following assumptions

- There exists a minimizer of f called x^*, for which $f^* = f(x^*)$
- There exists some bounding value G for which $\|g\|_2 \leq G, g \in \partial f(x), \forall\, x \in \mathbf{dom}\, f$[1][2]
- There exists some known bounding value R such that $R \geq \|x^{(1)} - x^*\|_2$

Next we can write the following based on (9)

$$\|x^{(k+1)} - x^*\|_2^2 = \|x^{(k)} - \alpha_k g^{(k)} - x^*\|_2^2 \tag{11}$$

Focusing on the right side, we can multiply the quadratic out

$$\|x^{(k+1)} - x^*\|_2^2 = \|x^{(k)} - x^*\|_2^2 - 2\alpha_k g^{(k)T}(x^{(k)} - x^*) + \alpha_k^2 \|g^{(k)}\|_2^2 \tag{12}$$

Using the definition of subgradient from (2), we can write

$$\|x^{(k+1)} - x^*\|_2^2 \leq \|x^{(k)} - x^*\|_2^2 - 2\alpha_k (f(x^{(k)}) - f^*) + \alpha_k^2 \|g^{(k)}\|_2^2 \tag{13}$$

[1]One way that this assumption can be met is if the function satisfies the Lipschitz condition, but it is not the only way

[2]This assumption helps us greatly in our analysis, but it is not strictly necessary. There exist subgradient methods which can be proven to work even when this assumption does not hold [1]

Paying particular attention to the term $\|x^{(k)} - x^*\|_2^2$, we can notice we have created a recursive inequality. If we perform a recursive substitution ultil reaching $x^{(1)}$ we would find ourselves with the following inequality

$$\|x^{(k+1)} - x^*\|_2^2 \leq \|x^{(1)} - x^*\|_2^2 - 2\sum_{i=1}^{k} \alpha_i (f(x^{(i)}) - f^*) + \sum_{i=1}^{k} \alpha_i^2 \|g^{(i)}\|_2^2 \tag{14}$$

We know that because of our bounding value $R \geq \|x^{(1)} - x^*\|_2$ and because of the norm, $\|x^{(k+1)} - x^*\|_2^2 \geq 0$, so we can write

$$2\sum_{i=1}^{k} \alpha_i (f(x^{(i)}) - f^*) \leq R^2 + \sum_{i=1}^{k} \alpha_i^2 \|g^{(i)}\|_2^2 \tag{15}$$

It follows readily from (10) that $f_{\text{best}}^{(k)} \leq f(x^{(i)}) \, \forall \, 1 \leq i \leq k$, so we can say

$$2\sum_{i=1}^{k} \alpha_i (f_{\text{best}}^{(k)} - f^*) \leq 2\sum_{i=1}^{k} \alpha_i (f(x^{(i)}) - f^*) \leq R^2 + \sum_{i=1}^{k} \alpha_i^2 \|g^{(i)}\|_2^2 \tag{16}$$

And therefore

$$f_{\text{best}}^{(k)} - f^* \leq \frac{R^2 + \sum_{i=1}^{k} \alpha_i^2 \|g^{(i)}\|_2^2}{2\sum_{i=1}^{k} \alpha_i} \tag{17}$$

We can finish by applying our G bounding condition to $\|g^{(k)}\|_2$, letting us write the final inequality

$$f_{\text{best}}^{(k)} - f^* \leq \frac{R^2 + G^2 \sum_{i=1}^{k} \alpha_i^2}{2\sum_{i=1}^{k} \alpha_i} \tag{18}$$

Given that R, G, and α_k are finite values, we can surmise that $f_{\text{best}}^{(k)}$ approaches f^* within some small error bound that is dependent on α_k and k. It is evident from this that the means we use to choose α_k will have significantly affect the error bound.

C. Choosing Step Size

There are many different methods of choosing the step size $\alpha^{(k)}$. We will be discussing five types in this section.

1) Constant Step Size: The first and simplest method we can use is a constant step size, where for some constant $\alpha > 0$

$$\alpha_k = \alpha \tag{19}$$

Given this, we can substitute (19) into (18) to obtain

$$f_{\text{best}}^{(k)} - f^* \leq \frac{R^2 + G^2\alpha^2 k}{2\alpha k} \tag{20}$$

To determine the converging behavior we can take a limit

$$\lim_{k \to \infty} \frac{R^2 + G^2\alpha^2 k}{2\alpha k} = \frac{G^2 \alpha}{2} \tag{21}$$

Therefore, we can surmise that we can reduce our eventual convergence bound by reducing α, although we should keep in mind that this comes at the expense of needing more steps. This is similar to the type of behavior we expect from other fixed step size methods.

2) Constant Step Length: Second is a constant step length, where for some constant $\gamma > 0$

$$\alpha_k = \frac{\gamma}{\|g^{(k)}\|_2} \tag{22}$$

It can be easily shown that for this step size, $\|x^{(k+1)} - x^{(k)}\|_2 = \gamma$, hence the name constant step length. Because $\|g^{(k)}\|_2 \leq G$, we can say that $\alpha_k \geq \gamma/G$, which enables us to use (17) to write

$$f_{\text{best}}^{(k)} - f^* \leq \frac{R^2 + \gamma^2 k}{2\sum_{i=1}^{k} \alpha_i} \leq \frac{R^2 + \gamma^2 k}{2\gamma k/G} \tag{23}$$

We can once more apply a limit to determine converging behavior

$$\lim_{k\to\infty} \frac{R^2 + \gamma^2 k}{2\gamma k/G} = \frac{G\gamma}{2} \tag{24}$$

3) Square Summable But Not Summable: This is a broad family of step size choosing methods, so long as the following are true

$$\alpha_k \geq 0, \|\alpha\|_2^2 = \sum_{k=1}^{\infty} \alpha_k^2 < \infty, \sum_{k=1}^{\infty} \alpha_k = \infty \tag{25}$$

Which, as the name implies, means that the squares are summable to a finite value but the values themselves are not. One typical example of this is $\alpha_k = a/(b+k)$ for some $a > 0$ and $b \geq 0$. Combining (18) and (25), we can derive

$$f_{\text{best}}^{(k)} - f^* \leq \frac{R^2 + G^2\|\alpha\|_2^2}{2\sum_{i=1}^{k} \alpha_i} \tag{26}$$

And once again using a limit to get the converging behavior

$$\lim_{k\to\infty} \frac{R^2 + G^2\|\alpha\|_2^2}{2\sum_{i=1}^{k} \alpha_i} = \frac{R^2 + G^2\|\alpha\|_2^2}{\infty} = 0 \tag{27}$$

Because $R^2 + G^2\|\alpha\|_2^2 < \infty$. Therefore $f_{\text{best}}^{(\infty)} \to f^*$, with no additional gap between them.

4) Nonsummable Diminishing Step Size: This is another broad family of step size choosing methods, where our criteria are as follows

$$\alpha_k \geq 0, \lim_{k\to\infty} \alpha_k = 0, \sum_{k=1}^{\infty} \alpha_k = \infty \tag{28}$$

As the name suggests, the step sizes eventually diminish to zero and the sum of all of the step sizes does not exist. One typical example is $\alpha_k = a/\sqrt{k}$ for some $a > 0$. Using (18) with (28) and taking limit, we find that the right hand size converges to zero because of the nonsummable denominator

$$\lim_{k\to\infty} \frac{R^2 + G^2\sum_{i=1}^{k} \alpha_i^2}{2\sum_{i=1}^{k} \alpha_i} = \frac{R^2 + G^2\sum_{i=1}^{k} \alpha_i^2}{\infty} = 0 \tag{29}$$

Therefore this type of method also guarantees the eventual convergence $f_{\text{best}}^{(\infty)} \to f^*$, with no gap between them.

5) Nonsummable Diminishing Step Length: Our final category of subgradient method is a step length method, similar to Section II-C2 in that we take the form $\alpha_k = \gamma_k / \|g^{(k)}\|_2$. Note that there is no longer a singular, constant variable γ, and instead we have a variable γ_k. These γ_k must fulfill the following critera

$$\gamma_k \geq 0, \lim_{k \to \infty} \gamma_k = 0, \sum_{k=1}^{\infty} \gamma_k = \infty \tag{30}$$

D. LASSO Regression

III. Simple Linear Example

A. Linear Function

To gain familiarity with subgradient methods, we can first implement a simple example of a linear piecewise function. This function is defined such that the objective function can be described as

$$f(x) = \max(Ax + b) \tag{31}$$

Where $A \in \mathbb{R}^{m \times n}$, $b_i \in \mathbb{R}^m$, and $\max$ picks the maximum value of of a vector. This function will have discontinuities where these linear functions meet. Also, being a pointwise maxiumum of a series of linear functions, this function will be convex. These traits together make this a suitable problem to practice subgradient methods on. In order to calculate a subgradient at a point x, we can find one of the constitutent functions that produces the maximum value and then its gradient will be a_i.

B. Source Code

Python was chosen to write a program to simulate this problem and solve it with an example of each of the 5 types of step size choice. It is broken into sections in this report and reformatted from its original source, but the original source code can be found along with all of the original files. See Appendix A.

1) Defining the problem: First we must define the dimensions of this problem $n = 10, m = 25$. Each of the linear equations was generated randomly using unit normal distrubutions, and by using enough seperate linear equations it was highly unlikely that the problem would end up unbounded below (During the testing this was an issue that was encountered).

```
#dimensions
n = 10
m = 25
#generate problem
a = np.random.normal(size=(m, n))
b = np.random.normal(size=m)
```

Next are the methods to determine the function value at a point and a subgradient at a point using the methods discussed in Section III-A

```
def f(x):
    values = np.matvec(a, x) + b
    return np.max(values)
def g(x):
    values = np.matvec(a, x) + b
    return a[np.argmax(values)]
```

C. Step Sizes

To use each of the five different types of step sizes, they were all implemented in their own methods. Where there was a typical example listed for a category, that example was used. In the case of nonsummable diminishing step lengths, there was no typical example provided in [1], so a step length version of the nonsummable diminishing step size was used. It fulfills the criteria in (30), and so is a valid representative of this type of method.

```
#step sizes
def constant_step_size(g, k):
    alpha = 0.01
    return alpha
def constant_step_len(g, k):
    gamma = 0.01
    norm = np.linalg.vector_norm(g)
    return gamma/norm
def square_sum_not_sum(g, k):
    a = 1
    b = 1
    return a/(b+k)
def nonsum_dim_size(g, k):
    a = 0.11
    return a / math.sqrt(k)
def nonsum_dim_len(g,k):
    a = 0.1
    gammak = a / math.sqrt(k)
    norm = np.linalg.vector_norm(g)
    return gammak/norm
```

D. Subgradient Method

This method uses whichever step size function is provided to execute a fixed number of steps, starting at the starting location.

```
def sgmethod(f, g, x0, step, maxiter):
    xkvec = [x0]
    xbest = x0
    for k in range(1,maxiter+1):
        xk = xkvec[-1]
        subgradient = g(xk)
        alphak = step(subgradient, k)
        xnext = xk - alphak*subgradient
        #keep track of best
        if f(xnext) < f(xbest):
            xbest = xnext
        xkvec.append(xnext)
    print(f"Finished. f* = {f(xbest):.5f}")
    return xkvec, xbest
```

E. Graphing

This final section of code calls the other methods, stores the appropriate data, and then generates the figures. Since the methods we are using do not guarantee finding the optimal f^* in a finite amount of time, the best value out of all of the values from all of the methods is taken as f^*.

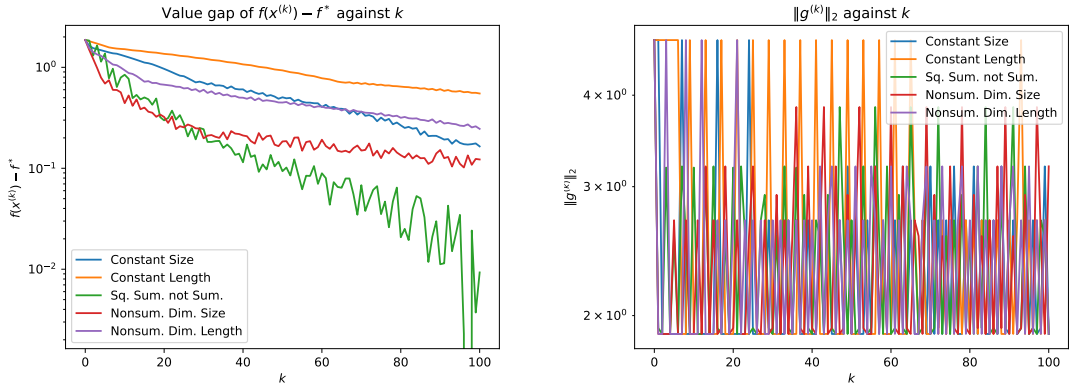
```
x0 = np.zeros(n)
maxiter = 100
fstar = math.inf
def format(name, step):
    global fstar
    xkvec, xbest = sgmethod(f, g, x0, step, maxiter)
    thisbest = f(xbest)
    if thisbest < fstar:
        fstar = thisbest
    fkvec = [f(xk) for xk in xkvec]
    sgvec = [np.linalg.vector_norm(g(xk)) for xk in xkvec]
    return name, xkvec, fkvec, sgvec

formatted = []
formatted.append(format("Constant Size", constant_step_size))
formatted.append(format("Constant Length", constant_step_len))
formatted.append(format("Sq. Sum. not Sum.", square_sum_not_sum))
formatted.append(format("Nonsum. Dim. Size", nonsum_dim_size))
formatted.append(format("Nonsum. Dim. Length", nonsum_dim_len))

karr = [k for k in range(0,maxiter+1)]
fig, ax = plt.subplots()
ax.set_title("Value gap of $f(x^{(k)}) - f^*$ against $k$")
ax.set_ylabel("$f(x^{(k)}) - f^*$")
ax.set_xlabel("$k$")
ax.set_yscale('log')
for entry in formatted:
    gapvec = entry[2] - fstar
    ax.plot(karr, gapvec, label=entry[0])
ax.legend()

fig2, ax2 = plt.subplots()
ax2.set_title("$\|g^{(k)}\|_2$ against $k$ with " + entry[0])
ax2.set_ylabel("$\|g^{(k)}\|_2$")
ax2.set_xlabel("$k$")
ax2.set_yscale('log')
for entry in formatted:
    ax2.plot(karr, entry[3], label=entry[0])
ax2.legend()

plt.show()
```



(a) Value gap plot

(b) Norm subgradient plot

Fig. 2: Plots for the linear problem

F. Figures

The two figures generated are Fig. 2a and 2b. We can make a few interesting observations that are congruent with what we have previously derived.

1) It is clear that none of these methods are descent methods, because they all increase at one point or another
2) Our last point is often not the most optimal point for a particular subgradient method

Additionally, we can note that at all points on all functions our subgradient normalized stuck to a few different magnitudes. It stands to reason that there would be exactly 25 such magnitues, each corresponding to one of our linear functions. In this case, it happened that the best performing method was using a square summable but not summable approach, however having run it a few times this is not always the case. This may be because of different function geometries behaving in different ways.

IV. LASSO Implementation

V. Numerical Results

VI. Conclusion

Appendix

A. Source Code Availablility

The source code for this project, including the code for this LaTeX document, is available on GitHub: https://github.com/Derpanieux/Convex-Optimization-Term-Project.

B. Proving Subgradient Properties

1) When the Function is Differentiable: We can begin by supposing that f is differentiable at x, and therefore $\nabla f(x)$ exists. Let us write a definition for each of its elements

$$\nabla f_i(x) = \lim_{y \to x} \frac{f(y) - f(x)}{y_i - x_i} \tag{32}$$

And let us also rewrite (3) with an elementwise sum

$$f(x) - f(y) + \sum_{i=1}^{n} g_i^T (y_i - x_i) \leq 0 \tag{33}$$

We can combine (32) and (33)

$$\lim_{y \to x} f(x) - f(y) + \sum_{i=1}^{n} \nabla f_i(x)^T (y_i - x_i) \leq 0 \tag{34}$$

Substitute and simplify and we find that

$$\lim_{y \to x} f(x) - f(y) + \sum_{i=1}^{n} \nabla f_i(x)^T (y_i - x_i) = 0 \tag{35}$$

If we are interested in trying an alternate $g \neq \nabla f_i(x)$, we can say that a particular g_i would be invalid if

$$g_i(y_i - x_i) > \nabla f_i(x)(y_i - x_i) \tag{36}$$

If $g_i > \nabla f_i(x)$ and $y_i > x_i$, this condition is fulfilled and therefore no values of $g_i > \nabla f_i(x)$ are valid. Likewise, if $g_i < \nabla f_i(x)$ and $y_i < x_i$ this condition is fulfilled and therefore no values of $g_i < \nabla f_i(x)$ are valid. Therefore the only valid values for every $g_i = \nabla f_i(x)$, so the only valid $g = \nabla f(x)$ if $\nabla f(x)$ exists.

2) Global Minimum: If x^* is a global minimum it must be true that $\forall\, y \in \mathbf{dom}\, f$

$$f(x^*) \leq f(y) \tag{37}$$

If we have g equal to the zero vector in (1), we end up removing the second term, so we end up with

$$f(y) \geq f(x) \tag{38}$$

We can trivially redefine $x = x^*$ to derive (37). Therefore x^* is optimal if and only if the zero vecctor is included in its subdifferential.

3) Convexity: We can view (3) as defining a halfspace in terms of x, y, f. From (4) we can see that $\partial f(x)$ is composed of intersections of these halfspaces. Since halfspaces are always convex and the intersection of convex sets is always convex, $\partial f(x)$ must be a convex set.

REFERENCES

[1] S. Boyd and J. Park, "Subgradient methods," May 2014. [Online]. Available: https://web.stanford.edu/class/ee364b/lectures/subgrad_method_notes.pdf

[2] S. Boyd, L. Xiao, and A. Mutapcic, "Subgradient methods," October 2003. [Online]. Available: https://web.mit.edu/6.976/www/notes/subgrad_method.pdf

[3] F. Santosa and W. W. Symes, "Linear inversion of band-limited reflection seismograms," *SIAM Journal on Scientific and Statistical Computing*, vol. 7, no. 4, pp. 1307–1330, 1986. [Online]. Available: https://doi.org/10.1137/0907087

[4] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 12 2018. [Online]. Available: https://doi.org/10.1111/j.2517-6161.1996.tb02080.x

[5] S. Boyd and L. Vandenberghe, "Subgradients," April 2008. [Online]. Available: https://see.stanford.edu/materials/lsocoee364b/01-subgradients_notes.pdf