A brief Review of ISTA and FISTA Algorithms in Sparsity-oriented Optimization.

Alan K. Nguyen, Shannon Kelley, Sudan Duwadi, Grant McConachie[a)]

(Dated: 30 September 2022)

This paper presents a detailed study of the Iterative Shrinkage-Thresholding Algorithm (ISTA) and its enhanced variant, the Fast Iterative Shrinkage-Thresholding Algorithm (FISTA), which both are helpful in solving optimization problems. ISTA combines gradient descent with shrinkage-thresholding to address problems in the form of the sum of a differentiable and a non-differentiable function, while FISTA builds on this by introducing an accelerated convergence mechanism, which helps increasing the efficiency while keeping ISTA's simplicity.

Keywords: ISTA, FISTA, Iterative Shrinkage-Thresholding Algorithm, Fast Iterative Shrinkage-Thresholding Algorithm.

I. INTRODUCTION TO ISTA

The Iterative Shrinkage-Thresholding Algorithm (ISTA) is a simple method used to solve optimization problems, especially those involving sparsity-inducing norms like the l_1-norm. The ISTA algorithm is designed to solve optimization problems of the form:

$$\min_{x \in \mathbb{R}^n} f(x) = g(x) + h(x) \tag{1}$$

where:

- $f(x)$ is the objective function.
- $g(x)$ is a smooth, convex function, typically representing a loss or data fidelity term.
- $h(x)$ is a non-smooth, convex function, often an l_1-norm, used to enforce sparsity in the solution.

A. ISTA Algorithm Steps

1. Initialization:

- Start with an initial guess x_0.
- Choose a step size $t > 0$.

Iteration: For each iteration $k = 0, 1, 2, \ldots$, the following steps are performed:

1. *Gradient Descent Step:*
   - Compute the gradient of the smooth part of the function $g(x)$ at the current estimate x_k.
   - Update the estimate using a gradient step:

$$y_{k+1} = x_k - t\nabla g(x_k) \tag{2}$$

   This step moves the estimate in the direction opposite to the gradient, aiming to reduce the value of the smooth part of the function.

[a)] Also at Biomedical Engineering Department, Boston University.

2. *Shrinkage-Thresholding Step:*

   - Apply the shrinkage-thresholding operator to handle the non-smooth part $h(x)$:

   $$x_{k+1} = S_t(y_{k+1}) \tag{3}$$

   - The shrinkage-thresholding operator $S_t(z)$ is defined as:

   $$S_t(z) = \text{sign}(z) \max(|z| - t, 0) \tag{4}$$

   where z can be either a scalar or a vector, and t is a non-negative threshold parameter. The operator applies component-wise when z is a vector. This operator enforces sparsity by shrinking elements of the estimate towards zero.

Convergence Check:

- The algorithm continues until a convergence criterion, such as a sufficiently small change in x_k, is met.

B. Sparsity and l_1-norm

Sparsity refers to the property of a vector xx having most of its elements equal to zero or close to zero. In other words, a sparse vector is one where only a few components are non-zero.

The l_1-norm of a vector x, denoted as $\|x\|_1$, is defined as the sum of the absolute values of its elements:

$$\|x\|_1 = \sum_i |x_i| \tag{5}$$

This norm is frequently used in optimization to promote sparsity in the solution vector x.

In optimization, maintaining sparsity means finding a solution where most components of x are zero or nearly zero. This is achieved by incorporating an l_1-norm term in the objective function, as in Lasso regression. The inclusion of this term encourages the solution to not only fit the data but also to have as many zero components as possible. It has some benefits such as:

1. **Feature Selection**: In machine learning, sparsity facilitates feature selection, identifying the most significant features (e.g. binary classification)
2. **Computational Efficiency**: Sparse representations often require less storage and processing power (digitally simple, reduce to 0's and 1's)

In conclusion, sparsity, promoted by the l_1-norm, plays a crucial role in various optimization problems by enabling simpler, more interpretable, and efficient solutions.

C. Shrinkage-Thresholding Operator $S_t(z)$

The shrinkage-thresholding operator $S_t(z)$ is a critical component in algorithms like the Iterative Shrinkage-Thresholding Algorithm (ISTA), where it enforces sparsity in optimization solutions.

Component-wise Operation For each component z_i of vector z, the operation is:

$$S_t(z_i) = \text{sign}(z_i) \max(|z_i| - t, 0) \tag{6}$$

Working Mechanism

1. **Magnitude Reduction:** The operator reduces the magnitude of each component of z by the threshold t, but not below zero. If $|z_i| > t$, it subtracts t from $|z_i|$; if $|z_i| \leq t$, it sets the component to zero.

2. **Sign Preservation:** The operator preserves the sign of each component.

Effect of the Operator

- **Sparsity:** By setting small components (absolute value less than t) to zero, the operator promotes sparsity in the solution.
- **Regularization:** It effectively implements l_1-norm regularization, known for its sparsity-inducing properties.

Example Consider $z = [3, -1, 0.5, -2]$ and $t = 1$. Applying $S_t(z)$ results in:

$$S_t(z) = [2, 0, 0, -1] \tag{7}$$

Components less than t are set to zero, enforcing sparsity.

D. Review of ISTA

Overview:
The Iterative Shrinkage-Thresholding Algorithm (ISTA) is an optimization algorithm used to find sparse solutions to problems where the objective function is the sum of a smooth, convex function and a non-smooth, convex function that encourages sparsity. ISTA is particularly useful in applications like compressed sensing, signal processing, and machine learning for feature selection.

Working Principle of ISTA

- **Gradient Descent on the Smooth Part:** ISTA performs a gradient descent step on the smooth part of the objective function to approach a minimum.
- **Shrinkage-Thresholding on the Non-Smooth Part:** It then applies a shrinkage-thresholding operation to the non-smooth part, promoting sparsity by reducing small values in the solution towards zero.

Basic Example Consider a simple linear regression problem with an l_1 regularization term (Lasso regression), where the objective is to minimize $f(x) = \|Ax - b\|_2^2 + \lambda\|x\|_1$, with A as a matrix, b as a vector, x as the solution vector, and λ as a regularization parameter.

1. **Initialization:** Start with $x_0 = 0$.

2. **Iteration:**

   - Perform a gradient descent step on $\|Ax - b\|_2^2$:

     $$y_{k+1} = x_k - t\nabla(\|Ax_k - b\|_2^2)$$

     where t is the step size.
   - Apply the shrinkage-thresholding operator for sparsity:

     $$x_{k+1} = S_{t\lambda}(y_{k+1})$$

     where $S_{t\lambda}(z) = \text{sign}(z)\max(|z| - t\lambda, 0)$.

3. **Repeat** until convergence.

In each iteration, ISTA moves closer to the solution that minimizes the overall objective, balancing fitting the linear model (represented by $Ax \approx b$) and maintaining sparsity in x.

II. FISTA REVIEWS

The Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) is an accelerated version of ISTA, designed to solve the same type of optimization problems but with significantly faster convergence.

A. FISTA's Acceleration Technique

FISTA introduces an acceleration step that significantly enhances the convergence rate compared to ISTA. Below, the key components of this technique:

- The update for x_k is given by the proximal map with respect to the step size L_t:

$$x_k = P_{L_t}(y_k) \tag{8}$$

  where P_{L_t} denotes the proximal operator associated with the Lipschitz constant L_t.

- The step size t_k is updated using the rule:

$$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2} \tag{9}$$

  This formula ensures that the sequence $\{t_k\}$ is monotonically increasing and contributes to the algorithm's acceleration.

- The auxiliary sequence y_k is updated based on the previous iteration's progress and the current step size:

$$y_{k+1} = x_k + \frac{t_k - 1}{t_{k+1}}(x_k - x_{k-1}) \tag{10}$$

  This equation introduces a 'momentum' effect, leveraging the progress made in the last iteration to anticipate the next step.
  This sequence is refined by a momentum-like term, which is a weighted combination of the current solution x_k and the progress made from the previous iterate x_{k-1}. The weighting factor $\frac{t_{k-1}}{t_{k+1}}$ plays a pivotal role in controlling the influence of the previous iteration's advancement on the next step. The main idea of FISTA lies in the self-contained update sequence $\{t_k\}$, which is recursively defined **solely dependent** on its preceding term t_{k-1}, entirely independent of any other parameters or smoothness constants of the functions $f(x)$ or $g(x)$. This increase convergence rate.

Convergence and Efficiency

Once again, the sequence $\{t_k\}$ is independent of the smoothness constant and derives its efficacy solely from the initial value t_1 and its own recursive definition. This self-contained sequence remarkably simplifies the convergence proof and enhances the convergence rate.

The vector y_k is considered 'progressive' as it is an extrapolation assuming the direction of the previous progress is correct, leading to a bolder step toward the solution compared to the conservative ISTA's update of x_k.

CONCLUSION

FISTA's acceleration technique improves upon ISTA by incorporating an extrapolation method that anticipates the next step, effectively reducing the number of iterations needed to reach a similar level of accuracy.

III. REFERENCES

[1]FISTA Algorithm, *CEREMADE, Dauphine*. https://www.ceremade.dauphine.fr/~carlier/FISTA
[2]FISTA Algorithm Explanation — Video, *YouTube*. https://www.youtube.com/watch?v=JRerBpNggN0
[3]ISTA Algorithm Inference, *Université de Sherbrooke*. https://info.usherbrooke.ca/hlarochelle/ift725/8_02_inference_ISTA_algorithm.pdf
[4]Sparse Signal Restoration Lecture Notes, *NYU*. https://eeweb.engineering.nyu.edu/iselesni/lecture_notes/sparse_signal_restoration.pdf