# Assignment 2: CS 754, Spring 2024-25


| Amitesh Shekhar | Anupam Rawat | Toshan Achintya Golla |
|---|---|---|
| IIT Bombay | IIT Bombay | IIT Bombay |
| 22b0014@iitb.ac.in | 22b3982@iitb.ac.in | 22b2234@iitb.ac.in |


February 16, 2025

**Declaration:** The work submitted is our own, and we have adhered to the principles of academic honesty while completing and submitting this work. We have not referred to any unauthorized sources, and we have not used generative AI tools for the work submitted here.

1. Your task here is to implement the ISTA algorithm:

   (a) Consider the 'Barbara' image from the homework folder. Add iid Gaussian noise of mean 0 and variance 4 (on a [0,255] scale) to it, using the 'randn' function in MATLAB. Thus $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{\eta}$ where $\boldsymbol{\eta} \sim \mathcal{N}(0, 4)$. You should obtain $\boldsymbol{x}$ from $\boldsymbol{y}$ using the fact that patches from $\boldsymbol{x}$ have a sparse or near-sparse representation in the 2D-DCT basis.

   (b) Divide the image shared in the homework folder into patches of size $8 \times 8$. Let $\boldsymbol{x_i}$ be the vectorized version of the $i^{th}$ patch. Consider the measurement $\boldsymbol{y_i} = \boldsymbol{\Phi x_i}$ where $\boldsymbol{\Phi}$ is a $32 \times 64$ matrix with entries drawn iid from $\mathcal{N}(0, 1)$. Note that $\boldsymbol{x_i}$ has a near-sparse representation in the 2D-DCT basis $\boldsymbol{U}$ which is computed in MATLAB as 'kron(dctmtx(8)',dctmtx(8)')'. In other words, $\boldsymbol{x_i} = \boldsymbol{U\theta_i}$ where $\boldsymbol{\theta_i}$ is a near-sparse vector. Your job is to reconstruct each $\boldsymbol{x_i}$ given $\boldsymbol{y_i}$ and $\boldsymbol{\Phi}$ using ISTA. Then you should reconstruct the image by averaging the overlapping patches. You should choose the $\alpha$ parameter in the ISTA algorithm judiciously. Choose $\lambda = 1$ (for a [0,255] image). Display the reconstructed image in your report. State the RMSE given as $\|X(:) - \hat{X}(:)\|_2 / \|X(:)\|_2$ where $\hat{X}$ is the reconstructed image and $X$ is the true image. Repeat this with the 'goldhill' image (take the top-left portion of size 256 by 256 only). [12 points]

   (c) Implement both the above cases using the FISTA algorithm from the research paper `https://epubs.siam.org/doi/10.1137/080716542`. [12 points]

   (d) Read the research paper and explain in which precise mathematical sense the FISTA algorithm is faster than ISTA. Also, why is it faster than ISTA? [12 points]

*Soln:*

(a) The Original and Noisy images(Gaussian noise of mean 0 and variance 4 was added) as as follows:

(a) Original Barbara

(b) Original Goldhill

Figure 1: Original Images

(a) Noisy Barbara

(b) Noisy Goldhill

Figure 2: Noisy Images

(b) The results of the ISTA reconstruction algorithm are as follows:

(a) Reconstructed Barbara(RMSE = 0.14021)

(b) Reconstructed Goldhill(RMSE = 0.28694)

Figure 3: Results of ISTA

(c) The results of the FISTA reconstruction algorithm are as follows:

(a) Reconstructed Barbara(RMSE = 0.10436)

(b) Reconstructed Goldhill(RMSE = 0.26047)

Figure 4: Results of FISTA

(d) The Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) provides a significant improvement over the standard Iterative Shrinkage-Thresholding Algorithm (ISTA) in terms of convergence speed. The primary reason for this acceleration is the incorporation of a momentum term inspired by Nesterov's acceleration method.
As explained in the paper, the convergence rate of ISTA is given by:

$$F(x_k) - F(x^*) \leq \mathcal{O}\left(\frac{1}{k}\right), \tag{1}$$

On the other hand, FISTA achieves a much faster convergence rate:

$$F(x_k) - F(x^*) \leq \mathcal{O}\left(\frac{1}{k^2}\right). \tag{2}$$

where $F(x)$ is the objective function, and $x^*$ is the optimal solution. This improvement implies that for the same number of iterations, FISTA gets significantly closer to the optimal solution than ISTA.

How do we explain this? The key idea behind FISTA's acceleration is the introduction of a momentum term that modifies the update step. Instead of using only the current iterate $x_k$, FISTA computes an auxiliary sequence $y_k$ as a linear combination of the current and previous iterates. For explanation using equations, consider the following:
Both ISTA and FISTA rely on the proximal operator $p_L(y)$, which is defined as:

$$p_L(y) = \arg\min_x \left\{ f(y) + \langle x - y, \nabla f(y) \rangle + \frac{L}{2}\|x - y\|^2 + g(x) \right\}. \tag{3}$$

Here, $p_L(y)$ finds the next iterate by combining a gradient descent step and a shrinkage/thresholding operation.
- In ISTA, the update is performed as:

$$x_{k+1} = p_L(x_k) \tag{4}$$

This means each step moves in the gradient direction while applying a thresholding operation.
- In FISTA, instead of applying $p_L$ directly to $x_k$, an intermediate variable $y_k$ is introduced to accelerate convergence:

$$x_{k+1} = p_L(y_k) \tag{5}$$

$$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2} \tag{6}$$

$$y_k = x_k + \frac{t_k - 1}{t_{k+1}}(x_k - x_{k-1}) \tag{7}$$

Calculating the momentum as shown above allows FISTA to "leap" forward in the optimization landscape, reducing the number of iterations required to reach a given accuracy.
FISTA is slightly more computationally expensive per iteration, but the faster convergence often makes it more efficient overall. Therefore, due to the improved convergence rate, FISTA is more efficient than ISTA in practical applications, particularly in image reconstruction and compressed sensing problems. The reduced number of iterations translates to significant computational savings while preserving the simplicity of ISTA.
In summary, in most practical scenarios, FISTA is the preferred choice due to its accelerated convergence and efficiency, although ISTA is simpler to implement and slightly more robust, making it a good choice for small-scale problems or when implementation simplicity is a priority.