Dynamical Characterization & Analysis of the Optimization Algorithms: Linearized Bregman and Iterative Shrinkage Thresholding Algorithm

Georgia Tech Mathematics College of Sciences

Ariba Khan, Yihua Xu, Rachel Kuske (Mentor)
Georgia Institute of Technology

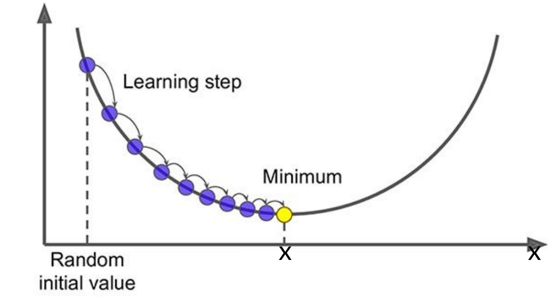
Abstract & Background

Sparse recovery optimization algorithms are utilized in machine learning, imaging, and parameter fitting in problems, as well as a multitude of other fields. Compressive sensing, a prominent field in mathematics this past decade, has motivated the revival of sparse recovery algorithms with ℓ-1 norm minimization. Although small underdetermined problems are substantially well understood, large, inconsistent, nearly sparse systems have not been investigated with as much detail.

In this dynamical study, two commonly used sparse recovery optimization algorithms, Linearized Bregman and Iterative Shrinkage Thresholding Algorithm are compared. The dependence of their dynamical behaviors on the threshold hyper-parameter and different entry sizes in the solution suggests complementary advantages and disadvantages. These results prompted the creation of a hybrid method which benefits from favorable characteristics from both optimization algorithms such as less chatter and quick convergence. The Hybrid method is proposed, analyzed, and evaluated as outperforming and superior to both linearized Bregman and Iterative Shrinkage Thresholding Algorithm, principally due to the Hybrid's versatility when processing diverse entry sizes.

	Linearized Bregman (LB)	Iterative Shrinkage Thresholding Algorithm (ISTA)
Iterative Formula	$z_{k+1} = z_k - t_k\, A^T\, (Ax_k - b)$ $x_{k+1} = S_\lambda\, (z_{k+1})$	$z_{k+1} = x_k - t_k\, A^T\, (Ax_k - b)$ $x_{k+1} = S_\lambda\, (z_{k+1})$
Time Step	$t_k = \dfrac{\lVert Ax_k - b\rVert_2^2}{\lVert A^\top(Ax_k - b)\rVert_2^2}.$	$t_k = \dfrac{1}{\lVert A\rVert_2^2}$
Shrinkage Process	$[S_\lambda(z_{k+1})]_i = \begin{cases} z_i - \lambda & \text{if } z_i > \lambda \\ 0 & \text{if } -\lambda \le z_i \le \lambda \\ z_i + \lambda & \text{if } z_i < -\lambda \end{cases}$	

Variable	Definition
k	Number of iterations
A	Usually a tall matrix
b	Multiplication of A and xTrue, with added noise
xTrue	The correct estimated value
Noise	Source of chatter
t_k	Time step
x_0	Initial guess for x



Sauggat Batharai 2018

Hyper - parameter Lambda

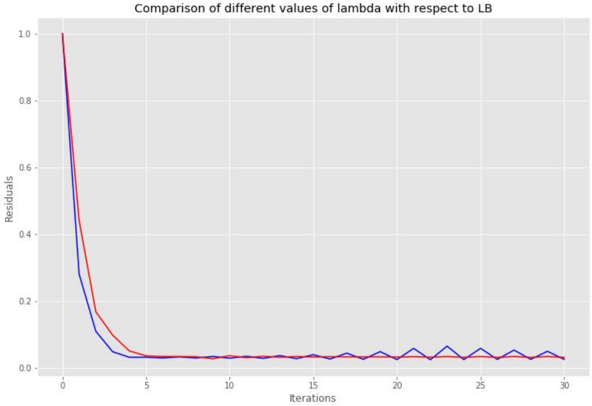
Ways of measurement

Normalized Residual

$$\mathcal{R}(x_k) = \frac{\|Ax_k - b\|_2}{\|b\|_2}.$$

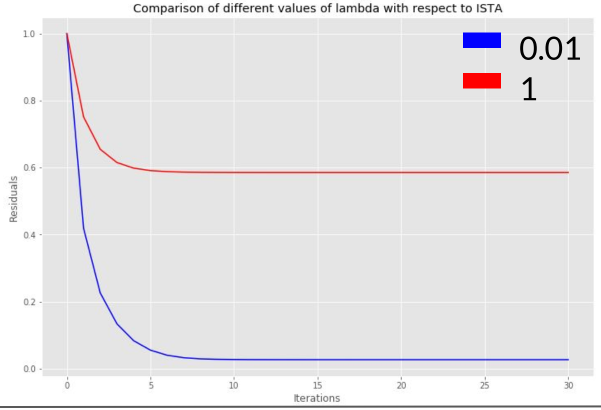
Model Error

$$M(x_k) = \frac{\|x_{True} - x_k\|_2}{\|x_{True}\|_2}$$



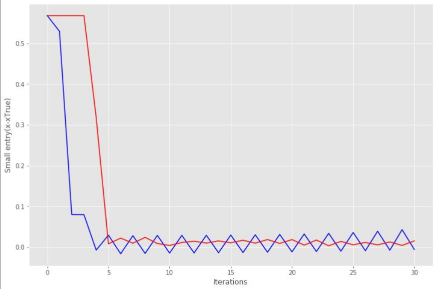
LB (Left): Though converging slightly quicker early on when λ is small, it would lead to overfitting/ fluctuation later on. Overall, a moderate or comparatively large λ is most suitable for LB, a reasonable range could be from 0.5 - 1.

ISTA (Right): Obviously, if we put a large λ in ISTA, the residual would decrease to a very limited extent. (Basically predicting most entries as zero). ISTA needs a very small λ to reach a low residual. A good value for λISTA is around 0.01.

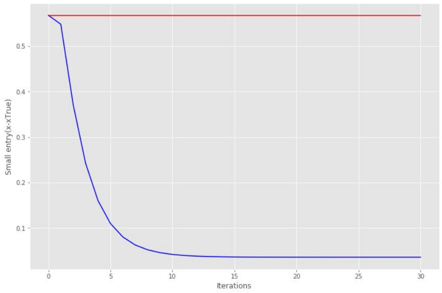


Small Entries

LB



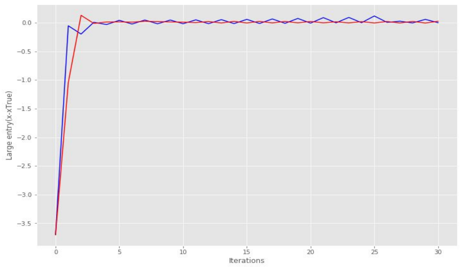
ISTA



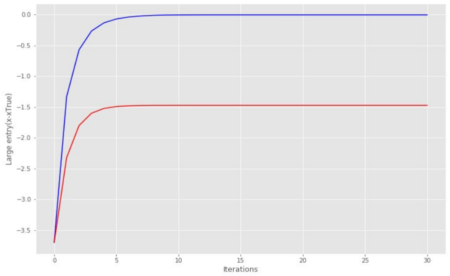
With small entries and small λ, ISTA tends to predict well and converge quickly without fluctuation in later iterations.

Large Entries

LB



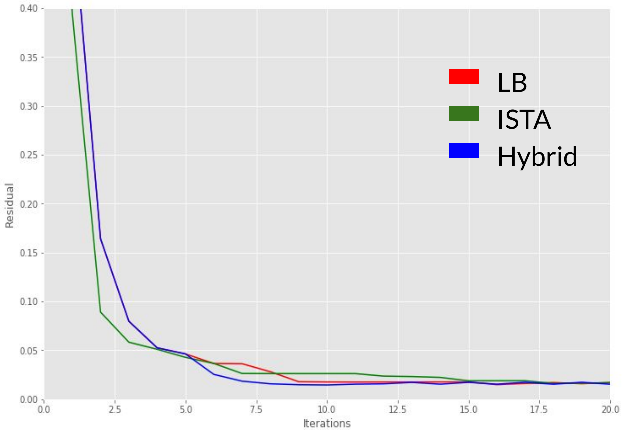
ISTA



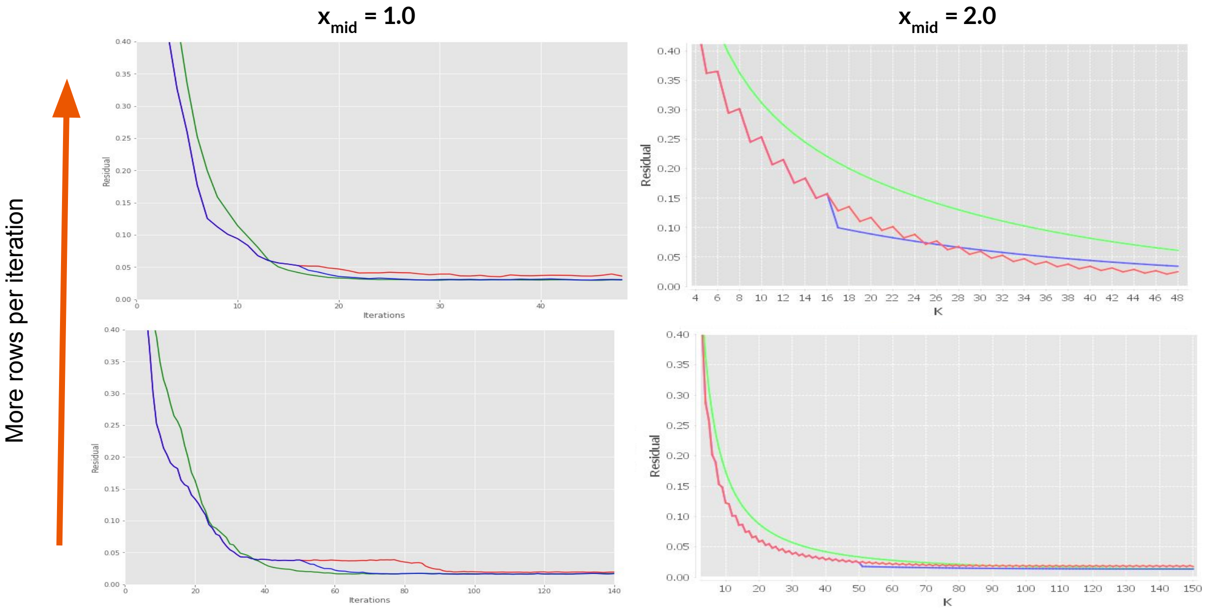
For large entries in LB, The difference between different λ is very small early on in the iterations and overall convergence rate is quick, while comparatively larger λ could result in smaller chatter later on.

The Hybrid Method

The Hybrid is a combination of both LB and ISTA algorithms. The Hybrid takes the advantages of quick convergence as well as less chatter by its ability to function well with both large and small entries. The general idea of the hybrid is as follows: x_{hybrid} follows x_{lb} for the first K iterations, then smaller entries follow ISTA and larger entries follow LB, where x is a specific entry.



In the figure (above), a base case scenario with the Hybrid method was run in which it was observed that the Hybrid method outperforms both LB and ISTA. Here the parameters were set as followed: $\lambda LB = 1$, $\lambda ISTA = 0.01$, $K = 5$, $K_{max} = 20$, and $x_{mid} = 1.0$. This observation prompted us to continue learning about the Hybrid method



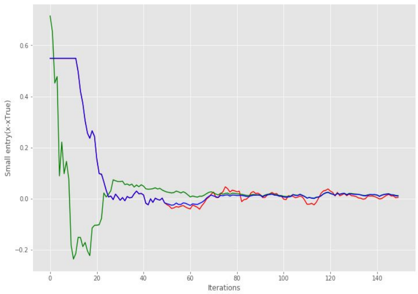
In the figure (left), all three algorithms underwent subsampling. In this study, subsampling is performed by choosing k rows out of the entire matrix A and vector b for each iteration which replaces the large matrix A and vector b with a subsampled A_k and b_k. Analytically, we predicted that LB will converge quicker than ISTA since ISTA leaves out information when undergoing the Shrinkage process because it uses x_k. The figures on the left support the prediction that LB converges quicker than ISTA when subsampled and the graphs also show that the Hybrid is able to outperform both LB and ISTA.

Comparing Entries & Submatrix Shapes

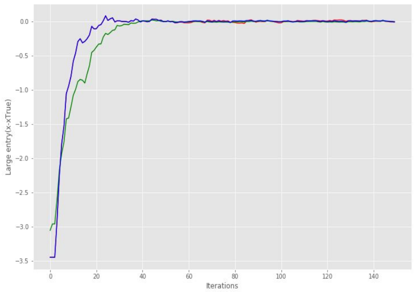
Georgia Tech Mathematics
College of Sciences

x-xTrue as all types of entries

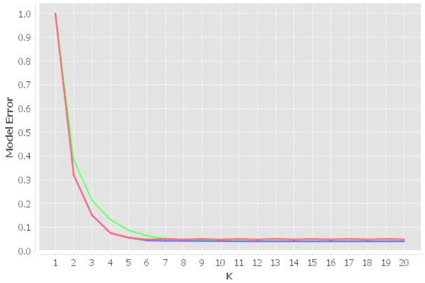
Small Entries

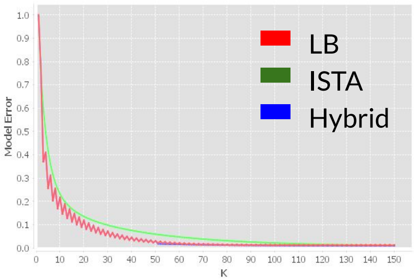


Large Entries

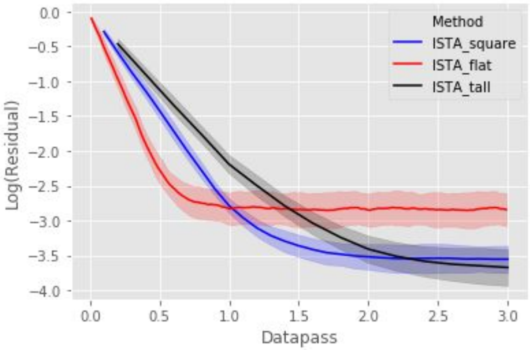


Model Error as one type of enty

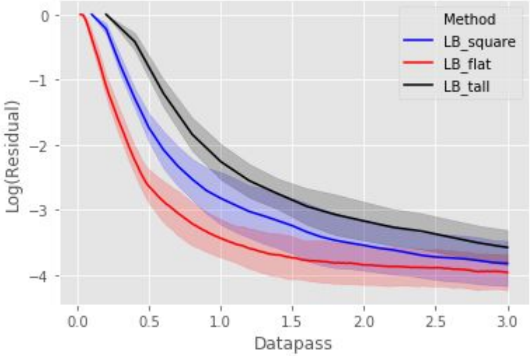




In figures (above), varying entry sizes were tested using an x-xTrue test and the Model Error test. The x-xTrue test follows a single entry size throughout the entire algorithm while in the Model Error test, all entries are set to be the desired size(zero,small, or large) and the behavior of all entries are tracked. Just as predicted earlier, the Hybrid suffers less chatter(an advantage from ISTA) and has a better approximation rate(an advantage from LB). For the zero entries, there is no observed significant difference with the x-xTrue test and the Model Error test is undefined since the denominator is zero. For small entries(entries less than one) we do observe a difference between the algorithms and we observe the Hybrid and ISTA working better than LB. Finally for the large entry column (entries greater than 1), we observe the Hybrid to be directly on top of the LB.



ISTA



LB

In figures (above), three subsampled matrix Ak shapes were tested (flat, tall, and square). For ISTA (top) a flat submatrix was observed to lead to overfitting in later iterations. The best submatrix for ISTA is Tall Subsampling with the least average residual. Conversely, for LB the best subsampling is flat subsampling since flat has the least residual. The y-axis is made to be log(Residual) to make the result more observable.