Proximal Gradient Algorithms for Robust Principal Component Analysis

Spandan Senapati(180782)
Computer Science and Engineering
Mode : Theory

***Abstract*— This paper depicts analysis of certain convex relaxation algorithms for solving the Robust Principal Component Analysis(RPCA) problem. The original formulation of the NP Hard problem can be formulated into a convex program. We analyse these algorithms, their convergence and possible modifications and experiments. Further we compare the performance of these algorithms on certain random datasets and analyse the convergence bounds.**

I. INTRODUCTION

RPCA is an inverse problem of recovering the low rank and sparse components given the data matrix. That is given a data matrix that could be grossly corrupted we wish to recover the low rank and the sparse components. A similar problem is SPCA where the data is also corrupted with a dense gaussian noise.

The convex problem we wish to solve for **RPCA** is

$$\min_{A,E\in\mathbb{R}^{m\times n}} ||A||_* + \lambda||E||_1 \text{s.t } A+E=D \tag{I.1}$$

Here $\|E\|_1 = \sum_{i,j}|E_{ij}|$ denotes the l_1 norm of a matrix and $\|A\|_* = \sum_{i=1}^{r}\sigma_i$ denotes the Nuclear norm of a matrix.

Similar variant for **SPCA** is

$$\min_{A,E,N\in\mathbb{R}^{m\times n}} |A||_* + \lambda||E||_1 \text{s.t} ||N||_F \leqslant \sigma, A+E+N=D \tag{I.2}$$

Unless mentioned we assume $||x||$ as the standard Euclidean Norm.

To ensure that the recovery problem makes sense certain incoherence conditions need to be satisfied. We state them directly as in [1]. A derivation of these can be obtained in [1]. We mention the result directly skipping the proof.

Throughout the analysis we assume that the optimisation problems we consider are feasible. Denote $(\hat{A},\hat{E})$ as the optimal solutions to I.1 Denote the singular value decomposition of $\hat{A} = U\Sigma V^T, U\in\mathbb{R}^{m\times r}, V\in\mathbb{R}^{n\times r}$, where r denotes the rank of $\hat{A}$. The incoherence conditions for $(\hat{A},\hat{E})$ for a parameter μ state that -

1) $\max_{i=1,2..m}||U^Te_i||^2 \leqslant \frac{\mu r}{m}, \max_{i=1,2..n}||V^Te_i||^2 \leqslant \frac{\mu r}{n}$
2) $||UV^T||_\infty^2 \leqslant \frac{\mu r}{mn}$

Similarly the incoherence conditions for $\hat{E}$ state that-

1) Denote the support set of $\hat{E}$ as $\Omega = \{(i,j)\text{s.t}\,\hat{E}_{ij}\neq 0\}$. Then entries in Ω are randomly distributed.
2) Entries in $\hat{E}$ are sampled from a uniform distribution and can be of large magnitudes and deviate in both directions. Mathematically this means $\hat{E}_{ij} \coloneqq \mathcal{U}[-\delta,+\delta]$ for some $\delta\in\mathbb{R}^{++}$.

This is where RPCA attains significant success compared to Classical PCA where even a fraction of the data is corrupted the estimates for the lower dimensional subspace are quite biased. In the problem of RPCA on the other hand relaxations on $\hat{E}$ lead to more robust algorithm.

Intuitively the incoherence conditions mean the low rank component should not be sparse and that the sparse component shouldn't be low rank.

The following theorem proved in [1] shows that choosing $\lambda = \frac{1}{\sqrt{\max\{m,n\}}}$ along with the incoherence conditions makes the convex program exactly solvable, i.e we can recover the optimal $(\hat{A},\hat{E})$.

Theorem I.1. *Suppose (A,E) satisfy the incoherence conditions and $\lambda = \frac{1}{\sqrt{\max\{m,n\}}}, n_1 = \max\{m,n\}, n_2 = \min\{m,n\}$ then with probabilty atleast $1-cn_1^{-10}$ for some c we have that $(A,E) = (\hat{A},\hat{E})$ provided the following conditions hold*

$$\textit{rank}(A) \approx \mathcal{O}(n_2(\log n_1)^{-2}) \quad \text{and} \quad ||E||_0 << n_1n_2 \tag{I.3}$$

where $||E||_0$ is the count of non zero entries in E.

The above result provides a strong theoretical bound on the convergence of various algorithms for our problem. We run our stimulations based on the theoretical bound on λ as given by the aforementioned result.

To solve **RPCA**(I.1) and **SPCA**(I.2) several approaches (both convex and non-convex) have been suggested such as ALM(Augmented Lagrange Multipliers) in [3], ASALM(Alternative splitting Augmented Lagrange Multipliers) and its variant VASALM in [5], ISTA(Iterative soft thresholding algorihm) and it's fast version(FISTA) in [7], some proximal gradient algorithms as in [8], [2], [4], ADMM and its variants etc. We restrict our analysis to convex relaxations of the problem (I.1).

One can broadly classify the existing lines of work along the following directions -

1) Proximal Gradient Methods - Unlike Classical GD, Proximal Gradient Methods optimise an objective function which corresponds to taking the Proximal

Map. However the computational bottleneck in these algorithms is whether the proximal map is analytically tractable.

2) Solving the Dual Problem via the augmented Lagrangian. After formulating the Lagrangian corresponding to the Dual Problem a class of algorithms follow the alternating optimisation method(namely ADMM). The computational expenses depend on the solutions of the subproblems on each of the optimisation variables.
3) Formulating the optimisation problem via smooth approximations to the objective functions.ALM and other variants follow such an approach. The idea is to approximate a non smooth convex function with a smooth function on which optimisation is rather simpler. Ex - $f_\mu(L) = \max_{\mathcal{W} \in \mathbb{R}^{m \times n}} \{\langle L, W \rangle - \frac{\mu}{2} \|W\|_F^2\} : \|W\| \leqslant 1$, where $\|W\|$ denotes the spectral norm of W, is a smooth approximation to $\|L\|_*$(conjugacy between the norms).

In this paper we provide an analysis of a fast proximal gradient algorithm as mentioned in [8] and see some variants that share a similar framework inspired by the the works of [7] and [6]

II. APGM

A. Motivation

1) Instability in Classical Gradient Algorithms - The iterate rule in classical GD is $x_k = x_{k-1} - \nabla f(x_{k-1})$. For RPCA we observe that a non smooth function like the nuclear norm doesn't have analytically tractable gradient updates and moreover with large matrices the computation of the the singular values becomes quite expensive. Hence there is quite some instabilty in the classical GD algorithms which are overcome by Proximal Gradient Algorithms which relax the problem by optimising a bound rather than the objective function

Inspired by the works of [9] we observe that judiciously choosing certain extragradient updates can greatly affect how fast convergence is attained.

B. Algorithm

Let our optimisation problem be $\phi = \min_x f(x) + \lambda g(x)$ where $f(x)$ is a smooth convex function that has a continuous Lipschitz gradient(we denote $L(f)$ as the smallest Lipschitz constant of $f(x)$) and $g(x)$ could be non - smooth(i.e nuclear norm). An approximation to the Upper Bound in Proximal Gradient Methods is -

$$Q(x, y) = f(y) + \langle x - y, \nabla f(y) \rangle + \frac{L}{2}||x - y||^2 + \lambda g(x)$$

It can be shown that $Q(x, y) \geqslant f(x) + \lambda g(x) \, \forall x, y \in \mathbb{R}^n \iff L \geqslant L(f)$ which is the **Quadriatic Bound Inequality**.

With the Quadriatic Approximation we can now run a Forward Backward(FB) splitting algorithm where we solve the optimisation problem - $\Psi : x_k = \underset{x}{\operatorname{argmin}} Q(x, x_{k-1})$. However this leads to a suboptimal convergence rate of $\mathcal{O}(1/k)$. However choosing suitable extragradient rules can lead to an optimal convergence rate of $\mathcal{O}(1/k^2)$ as has been shown by [9] for general convex programming problems. The fast version of PGM has the update rule

1) $y_k = x_k + \frac{(t_{k-1}-1)}{t_k}(x_k - x_{k-1})$
2) $x_{k+1} = \underset{x}{\operatorname{argmin}} Q(x, y_k)$. The separable structure of the problem allows us to write $\operatorname{argmin}_x Q(x, y_k) = \operatorname{argmin}_x \left(\lambda g(x) + \frac{L}{2}||x - (y - \frac{1}{L}\nabla f(y))||^2 \right)$
3) $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$

While $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$ works really well in practice we ran stimulations on different functions of t as

$$f(t_k) = \frac{1 + t_k}{2}, f(t_k) = \frac{1 + \sqrt{1 + t_k^2}}{2}$$

These values worked well in practice however the convergence wasn't affected by significant margins. [9] showed that $t_{k+1} \leqslant \frac{1+\sqrt{1+4t_k^2}}{2}$ ensures convergence in general in proximal gradient algorithms.

C. Analysis

To sequence of values produced by classical Proximal Gradient Method $x_k = \underset{x}{\operatorname{argmin}} Q(x, x_{k-1})$ are monotonically decreasing. We observe that for $x = y \implies Q(x, y) = f(x) + \lambda g(x) = F(x)$. Further the optimisation steps are such that $Q(x_k, x_{k-1}) \leqslant Q(x_{k-1}, x_{k-1})$. It follows by the Quadriatic Bound Inequaity(with $L \geqslant L(f)$) that -

$$F(x_k) \leqslant Q(x_k, x_{k-1}) \leqslant Q(x_{k-1}, x_{k-1}) = F(x_{k-1})$$

Relaxing the equality constraint **RPCA** can be expressed as an unconstrained optimisation problem with relaxation factor μ as -

$$\underset{A,E}{\operatorname{argmin}} F(A, E) = \mu(||A||_* + \lambda||E||_1) + \frac{1}{2}||D - A - E||^2$$

Under the PGM framework inducing the QUB Inequality and auxiliary variables (Y_A, Y_E) the subproblems we wish to solve for iterate updates become separable. These are the corresponding **Proximal** Maps corresponding to (A, E) -

$$P : \operatorname{argmin}_A \mu||A||_* + \frac{L}{2}\left\|A - \left(Y_A - \frac{1}{L}\left(Y_A + Y_E - D\right)\right)\right\|^2$$

$$Q : \operatorname{argmin}_E \mu\lambda||E||_1 + \frac{L}{2}\left\|E - \left(Y_E - \frac{1}{L}\left(Y_A + Y_E - D\right)\right)\right\|^2$$

Where we denote $G_A = \left(Y_A - \frac{1}{L}\left(Y_A + Y_E - D\right)\right)$.Fortunately from [10] we observe that P and Q can be solved with closed form solutions given by

$$\boxed{E^* = \mathcal{T}_{\frac{\lambda\mu}{L}}\left(Y_E - \frac{1}{L}\left(Y_A + Y_E - D\right)\right)}$$

and with the SVD decomposition of $G_A = U\Sigma V^T$ as

$$\boxed{A^* = U\mathcal{T}_{\frac{\mu}{L}}(\Sigma)V^T}$$

where $\mathcal{T}_\epsilon(x) = \operatorname{sgn}(x)\max(|x| - \epsilon, 0)$ denotes the soft thresholding operator.

D. Convergence

APGM and Iterative Thresholding Algorithm(FISTA) share a similar framework in their iterate updates and hence their convergence bounds.

Functional Iterates - APGM has been shown to have a functional convergence rate of $\mathcal{O}(1/k^2)$. [9] and [7] provide theoretical bounds on the convergence of the functional values and [6] on the weak convergence of the optimisation variables. For some C at the end of iteration k,

$$F(A_k, E_k) - F(\hat{A}, \hat{E}) \leqslant \frac{C}{k^2}$$

The constant depends on the explicit time update steps.

E. Time Complexity and Possible Issues

1) For **RPCA**, $L_f = 2$ and hence the significant computation goes into the SVD decomposition in each iteration. Each iteration takes $\mathcal{O}(\max(m,n))^3$. Thus for large matrices this becomes computationally expensive.
2) In problems where the computation of L_f becomes expensive we tried out the suggested *Backtracking* approach as mentioned in [7]. Further we tried *Randomized Determinant* estimation algorithms for computing the Lipschitz constant in linear inverse problems.

III. ISTA AND FISTA

A. Introduction

ISTA and FISTA share a similar framework with the APGM Algorithm discussed above and find quite a lot of applications in image deblurring, compressed sensing problems. In this section we mostly emphasise on the theoretical bounds concerning convergence and an analysis of backtracking step. Our stimulations showed that FISTA had superior convergence compared to ISTA and state of art exisiting optimizers in Pytorch's optim library. The optimisation problem we wish consider is -

$$P : \text{argmin}_x F(x) = f(x) + \lambda g(x)$$

where similar assumptions on continuity and smoothness are made as in section on APGM. We denote $L(f)$ as the smallest Lipschitz constant of $f(x)$. For analysis and implementation purposes we consider $f(x) = \|\mathcal{A}x - b\|^2$ as in [7]. $Q(x,y)$ is defined as in APGM as -

$$Q(x,y) = f(y) + \langle x - y, \nabla f(y)\rangle + \frac{L}{2}||x-y||^2 + \lambda g(x)$$

B. Background

The following Lemma stated as QUB in the above section leads to motivation towards the Backtracking approach in FB algorithms.

Lemma III.1. *For all $x, y \in \mathbb{R}^n$ we have*

$$F(x) \leqslant Q(x,y) \iff L \geqslant L(f)$$

where $L(f)$ denotes the smallest Lipschitz constant of $f(x)$

Proof

- The forward implication is standard and can be found in classical GD literature. An interesting Geometric intuition concerning the reverse implication is provided here.
- The following equivalence can be established using Cauchy-Schwarz inequality and Lipschitz continuity for $f(x)$

  $$F(x) \leqslant Q(x,y) \iff \langle y - x, \nabla f(y) - \nabla f(x)\rangle \leqslant L||x-y||^2$$

  We show that it's not possible to have $L < L(f)$. Define $b = y - x, a = \nabla f(y) - \nabla f(x)$. So we have $||a|| \leqslant L(f)||b||$ (Lipschitz condition). WLOG assume $||b|| = 1$.
  Hence we want $\forall\, a, b \in \mathbb{R}^n$

  $$\underbrace{a^T b}_{\text{projection of a on b}} \leqslant L, ||a|| \leqslant L(f)$$

  Clearly $a \in \mathcal{B}_c(0, L(f))$ represents a norm ball centered at origin. We observe that the locus

  $$a \text{ s.t } \left\{ a^T b \leqslant L, L < L_f, ||a|| < L(f) \right\} \neq \mathcal{B}_c(0, L(f))$$

  thereby posing a contradiction. Hence $L \geqslant L_f$.

C. Analysis

For $f(x) = \|\mathcal{A}x - b\|^2$ we observe that $L(f) = 2\lambda_{\max}(A^T A)$. Assuming the computation is tractable using the framework of APGM the updates in ISTA become $x_k = \text{argmin}_x Q(x, x_{k-1})$. The update rules for FISTA are similar to Fast PGM which was discussed in section on APGM. These are the constant step size approach as the L that's used in the algorithm is known from the computed value $L(f) = 2\lambda_{\max}(A^T A)$.

Backtracking - With $L(f)$ being difficult to compute for large matrices, a backtracking approach has been suggested in [7]. The method isn't specific to solving inverse convex programming problems but in the general framework where we wish to obtain a reasonable estimate that works well in optimal/suboptimal complexity such as in Inverse estimation of large matrices etc.

Analysis of Backtracking - We use the following notation for the subproblem that is solved in general proximal gradient algorithm. This corresponds to the proximal map for y as -

$$\begin{aligned} p_L(y) &= \text{argmin}_x\ Q(x,y) \\ &= \text{argmin}_x\ \lambda g(x) + \frac{L}{2}\left\| x - \left(y - \frac{\nabla f(y)}{L} \right) \right\|^2 \end{aligned}$$

Start with a $L_0 > 0$, hyperparameter η. At every iteration k we find some smallest $L_k = \eta^{i_k} L_{k-1}$ such that $F(p_{L_k}(x_{k-1})) \leqslant Q(p_{L_k}(x_{k-1}), x_{k-1})$ and set $x_k = p_{L_k}(x_{k-1})$. Note that QUB holds for $L \geqslant L(f)$ and for any value of $L < L(f)$ it may or may not hold thereby asserting the validity of the backtracking step. However once $L_k \geqslant L(f)$, L_k remains fixed as QUB starts to hold further and the backtracking step holds trivially. Hence there exists some small constant $c = \mathcal{O}(\eta)$ s.t $L_k \leqslant cL(f)$. In every

iteration we essentially push the current L towards $L(f)$ by scaling it with η.

The loss decreases as in APGM(but not due to QUB but the backtracking step) as -

$$\underbrace{F(x_k) \leqslant Q(x_k, x_{k-1})}_{\text{Backtracking step}} \leq Q(x_{k-1}, x_{k-1}) = F(x_{k-1})$$

Randomized Estimators [11] proposed randomized algorithms for estimation of the determinants and trace of matrices. Following their approach we tried estimating the maximum eigenvalues and ran stimulations with $L = L_{\text{est}}$. Convergence was well attained with the estimated values for matrices well within $n \approx 600$. We plot the relative errors in approximations $\Delta = \frac{|L_{\text{act}} - L_{\text{ets}}|}{L_{\text{act}}}$.$L_{\text{act}} = 2\lambda_{\max}(A^T A)$

D. Convergence Analysis

In general Forward Backward splitting methods like ISTA obey convergence given by - $F(x_n) - F(\hat{x}) \leqslant \frac{C}{n}$.A derivation of this can be found in [12].Further the sequence of iterates $(x_n)_{n\in\mathbb{N}}$ weakly converges. For accelerated FB algorithms like FISTA the bound can be shown to be $\frac{C}{n^2}$. A proof of the same can be obtained in [7] with their more general result being -

$$\boxed{F(x_n) - F(\hat{x}) \leqslant \frac{1}{2\gamma t_n^2} \|x_0 - \hat{x}\|^2}$$

where $\gamma \leqslant \frac{1}{L(f)}$ and $\hat{x}$ is an optimal solution .Further it's trivial by induction that $t_n \geqslant n$ hence giving a bound of $\frac{C}{n^2}$. Note that C depends on the explicit function t_n.

The Case of weak convergence of $(x_n)_{n\in\mathbb{N}}$ is however more interesting. [6] present an excellent analysis of this. We summarise important results of their proof -

1) Their analysis considers a specific class of functions $t_n = \frac{n+a-1}{a}$ where a is some constant.
2) While $t_{n+1}^2 - t_{n+1} - t_n^2 = 0$ ensures tight convergence bounds, other functions like $t_n = \frac{n+a-1}{a}$ ensure better global properties of the sequence $nw_n = n(F(x_n) - F(\hat{x})$, That is $\liminf_{n\to\infty} n^2 \log n w_n = 0$.
3) Boundedness of x_n. Follows from the boundedness of v_n as **Lemma 2** in [6] states that -

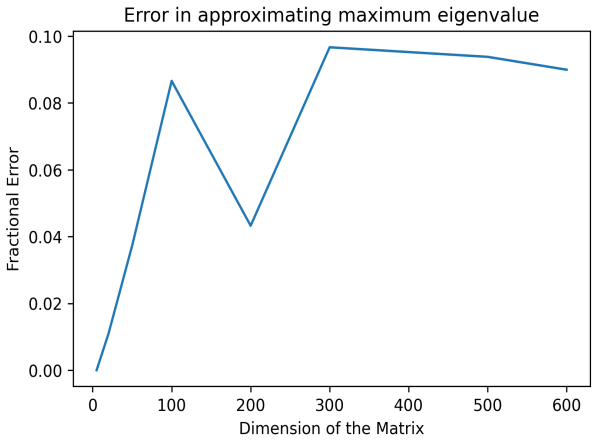
$$t_{n+1}^2 w_{n+1} + \sum_{n=1}^{N} \rho_{n+1} w_n \leqslant \frac{\nu_0 - \nu_{N+1}}{\gamma} \implies$$

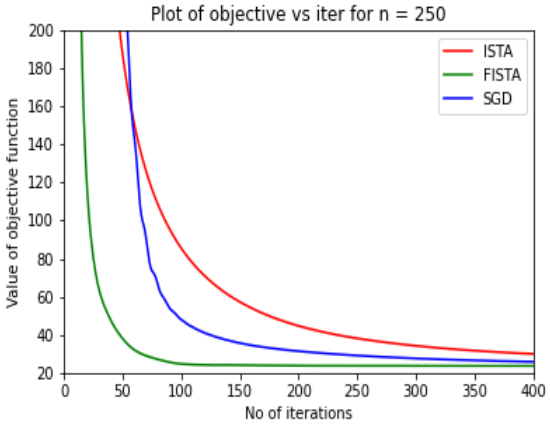
$$\nu_{N+1} \leqslant \nu_0 - \gamma\left(t_{n+1}^2 w_{n+1} + \sum_{n=1}^{N} \rho_{n+1} w_n\right)$$

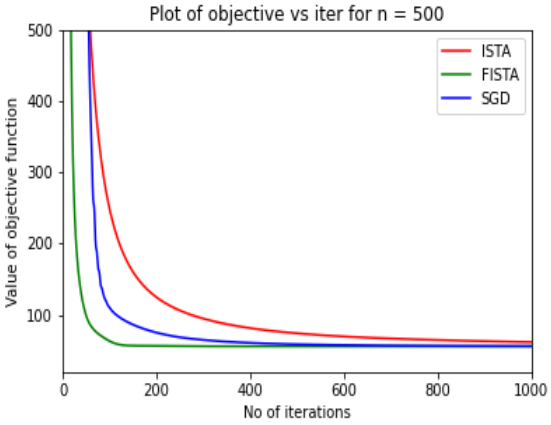
where $\rho_n = t_{n-1}^2 - t_n^2 + t_n, \nu_n = \frac{1}{2}\|u_n - \hat{x}\|^2, u_n = x_{n-1} + t_n(x_n - x_{n-1}), w_n = F(x_n) - F(\hat{x})$ as in [6] thereby implying boundedness of ν_n which in turn implies boundedness of u_n and hence x_n

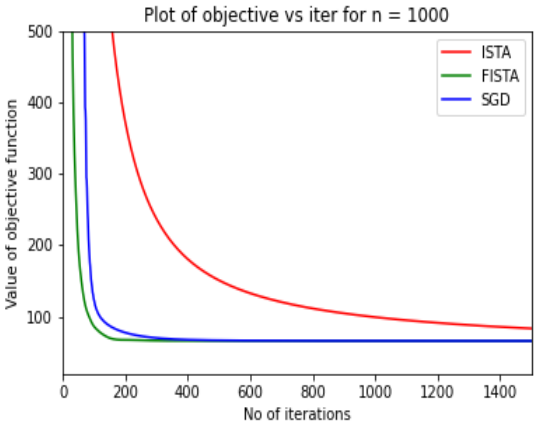
4) The final step is analysing the convergence of $x_n$to a fixed minimzer $\hat{x}$ which [6] do through the convergence of $\Phi_n = \|x_n - \hat{x}\|$.

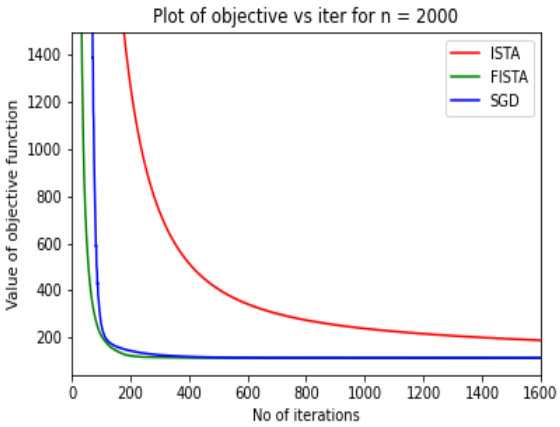
Comparison of further stimulations of $t_n = \frac{n+a-1}{a}$ and $t_n = \frac{1+\sqrt{1+4t_n^2}}{2}$ on **RPCA** can be found in [6].

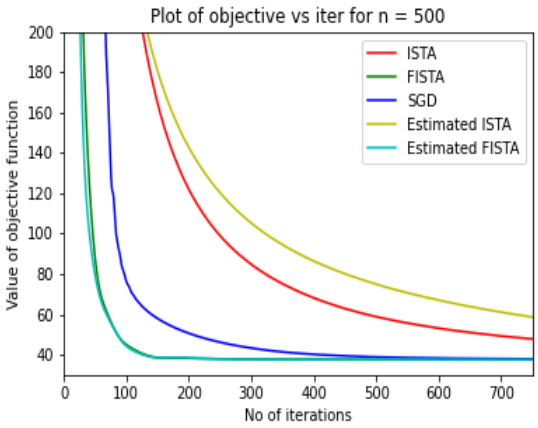


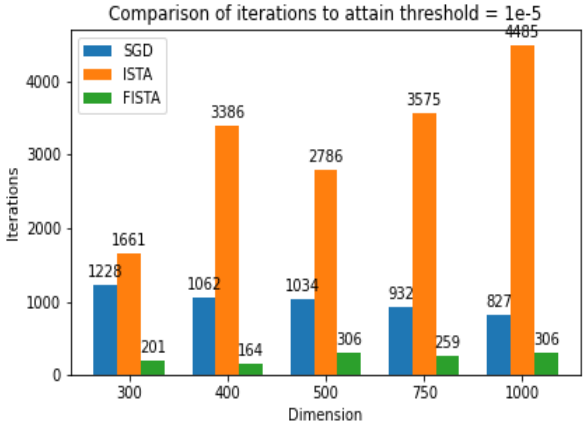


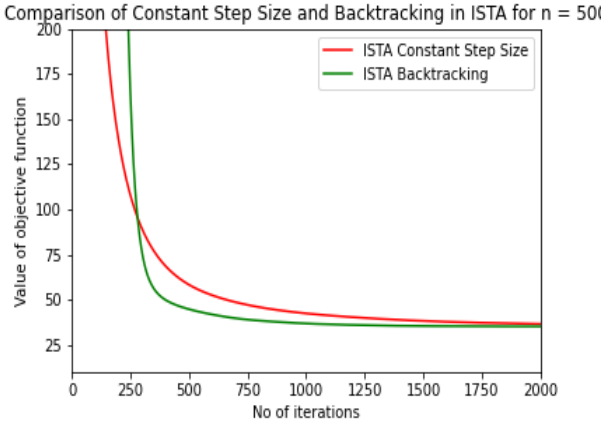












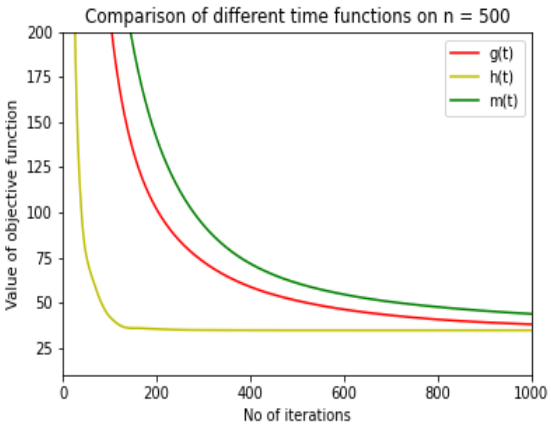


Fig. 1. $g(t) = \frac{1+\sqrt{1+t^2}}{2}, h(t) = \frac{1+\sqrt{1+4t^2}}{2}, m(t) = \frac{1+t}{2}$

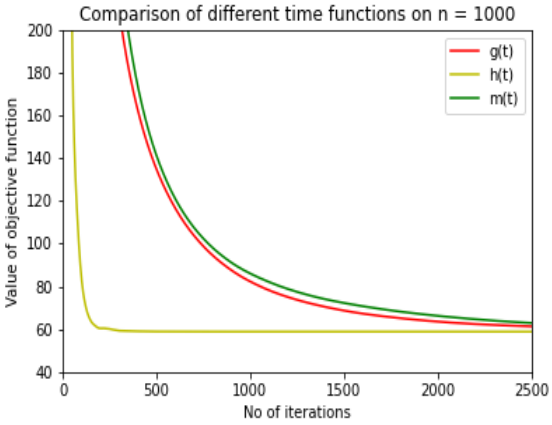
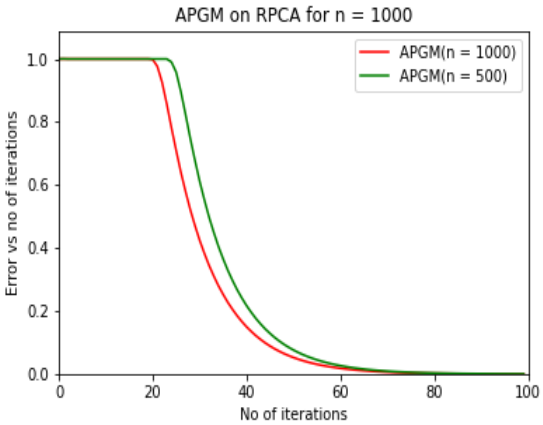


Fig. 2. $g(t) = \frac{1+\sqrt{1+t^2}}{2}, h(t) = \frac{1+\sqrt{1+4t^2}}{2}, m(t) = \frac{1+t}{2}$



IV. EXPERIMENTS

We carried stimulations on APGM and the FISTA framework. A detailed comparison of the performance of thresholding algorithms and APGM can be found in [8]. We summarise the results of our stimulations as follows -

A. APGM

We created datasets for **RPCA** in a similar manner as stated in [2] as -

1) Denote $\hat{A} = UV^T$, $U \in \mathbb{R}^{n\times r}$, $V \in \mathbb{R}^{r\times n}$, where r is the rank of the matrix($r = c_r n$). $U_{ij} \sim \mathcal{N}(0,1), V_{ij} \sim \mathcal{N}(0,1)$ where all U_{ij}, V_{ij} are independent variables.
2) The non zero entries in $\hat{E}$ are chosen randomly and independently from a uniform distribution $\mathcal{U}\left(-\sqrt{\frac{8r}{\pi}}, \sqrt{\frac{8r}{\pi}}\right)$. The cardinality $|\Omega| = c_p n^2$
3) The error at the end of k^{th} iteration is calculated as $\text{Err} = \frac{\left\|A_k - \hat{A}\right\|_F}{\left\|\hat{A}\right\|_F}$. We used $c_r = 0.05, c_p = 0.05$ in our stimulations, and $\lambda = \frac{1}{\sqrt{n}}$ for objective in **RPCA**

There wasn't any significant leap on the performance using different time functions and the randomized estimators for $\lambda_{\max}$. We found that the theoretical bound as mentioned in **Theorem I.1** works really well in practice.

B. ISTA and FISTA

We considered $f(x) = \|\mathcal{A}x - b\|^2$ and compared the performances of three different optimisation algorithms - *SGD, ISTA, FISTA* considering the following measures -

1) Iterations to attain convergence within a threshold bound of $|F(x_k) - F(x_{k-1})| \leqslant \epsilon = 10^{-5}$ and no. of iterations taken in general for convergence.
2) Comparison of performances considering different time step updates.
3) Performances considering randomized estimators for $L(f)$ as mentioned in [11]. The labels *Estimated FISTA* and *Esimated ISTA* in the curves denote stimulations with randomized estimators. The results were not much affected as the relative error in approximation is generally small $\Delta \approx 0.05$.
4) Performances considering Backtracking and constant step size approach. We found that the progress in the backtracking approach are highly dependant on the initial estimates A_0 and the hyperparameter η and are much more prone to instabilities and fluctuations.

The results of experiments can be found in the attached figures.

REFERENCES

[1] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" Journal of ACM, vol. 58, no. 1, pp. 1–37, 2009.

[2] Shiqian Ma and Needat Serhat Aybat, "Efficient Optimization Algorithms for Robust Priincipal Component Analysis and Its Variants"

[3] Z. Lin, M. Chen, L. Wu, and Y. Ma, "The augmented lagrange multiplier convex optimization algorithms for exact recovery of a corrupted low-rank matrices." UIUC Technical Report UILU-ENG-09-2215, Tech. Rep., 2009.

[4] N. S. Aybat, D. Goldfarb, and S. Ma, "Efficient algorithms for robust and stable principal component pursuit problems," Computational Optimization and Applications, vol. 58, pp. 1–29, 2014.

[5] M. Tao and X. Yuan, "Recovering low-rank and sparse components of matrices from incomplete and noisy observations," SIAM Journal on Optimization, vol. 21, no. 1, pp. 57–81, 2011.

[6] Antonin Chambolle, Charles Dossal. On the convergence of the iterates of "FISTA". Journal of Optimization Theory and Applications, Springer Verlag, 2015, Volume 166 (Issue 3), pp.25. hal- 01060130v3

[7] "A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems", Amir Beck and Marc Teboulle, SIAM J. I MAGING S CIENCES Vol. 2, No. 1, pp. 183–202.

[8] Z. Lin, A. Ganesh, J. Wright, L. Wu, M. Chen, and Y. Ma, "Fast convex optimization algorithms for exact recovery of a corrupted low-rank matrix," in International Workshop on Computational Advances in Multi-Sensor Adaptive Processing, 2009.

[9] Y. Nesterov, "A method for solving covex prorgamming problems with a convergence of $\mathcal{O}(1/k^2)$".

[10] "G.A. Watson", "Characterization of the subdifferential of some matrix norms".

[11] A randomized algorithm for approximating the log determinant of asymmetric positive definite matrix - Christos Boutsidis, Petros Drineasa, Prabhanjan Kambadurb, Eugenia Maria Kontopouloua, Anastasios Zouzias

[12] Yurii Nesterov. Smooth minimization of non-smooth functions. Math. Program., 103(1, Ser. A):127–152, 2005