10/10

Reading Report of: *Interior-Point Method for Large-Scale ℓ_1-reg. LS* by Kim et al.

(ECE 695, Reading Assignment 01, Spring 2018)

Overview of the Paper

The problem at hand is the well known least squares problem with ℓ_1 regularization as in equation 3 of the paper . The methods in this paper attempt to find the solution faster than previous methods or with higher numerical accuracy while allowing dense data matrices. The problem is compared to its ℓ_2 -regularized counter part which can be solved in closed form partly due to its differentiability and strict convexity. The ℓ_1 cost function is not differentiable and thus is solved numerically using a Newton method based approach that uses fast algorithms for the arising subproblems.

Further, the performance of this method is demonstrated on numerical examples while the computational complexity is empirically determined. Also, possible extensions and variations are described.

Prior Work

For problem this there were a wide variety of algorithms known. As generic algorithms for non-differentiable objective functions are often slow, some methods aim to exploit certain prior assumptions about the problem. Examples are path-following-methods that are suitable for extremely sparse solutions. Also, algorithms that used specialized specialized matrix-vector operations can be suitable for large problems. Further, a variety of iterative gradient descent or coordinate descent methods were shown to handle larger problems efficiently.

Key Ideas of the Paper

To deal with the non-differentiability of the primal cost function the problem is transformed into a differentiable minimization with linear inequality constraints, shown in equation 13 of the paper:

$$\text{minimize} \quad \phi(x,u) = ||Ax-y||_2^2 + \lambda \sum_i u_i \qquad \text{subject to} \ -u_i \leq x_i \leq u_i \text{ for all } i \tag{1}$$

The constraints are then lifted by augmenting the objective function of equation 13 (1) with a logarithmic barrier function $\Phi(x,u)$. The barrier is smooth, convex and grows unbounded at points near the boundaries of the feasible set. A minimization of $\phi_t(x,u) = t\phi(x,u) + \Phi(x,u)$ is simple (when given a feasible starting point) as it is strictly convex and differentiable. However, a minimizer, $(x(t), u(t))$, of $\phi_t(x,u)$ is not not necessarily primal feasible for any t. Also, note that for $t = \infty$ the minimizer, $x(t)$, is feasible and $x(t) = x^\star$ is a solution to the overall primal problem as the barrier function $\Phi(x,u)/t$ resembles an (infinitely high) indicator function for the feasible set when $t \to \infty$. However, as the minimization for large (or infinite) t is very difficult the proposed method aims to start at a feasible suboptimal point and solves the minimization of $\phi_t(x,u)$ for increasing t in an iterative fashion.

To solve the resulting convex optimization for a given t, a Newton based method is applied where the solution of Newton system from equation 14 in the paper yields the search direction $(\Delta x, \Delta u)$. The key ideas to achieve the performance are:

- only approximately solving the Newton system using PCG with an good initial point and and an efficient stopping criterion (truncation rule)

- performing a fast line search using the described backtracking method

- using a good update scheme for the t-variable that balances the trade-off between "*fast minimization with Newton*" vs. "*fast closing of the duality gap*".

Comments

As I am completely new to ℓ_1-regularized LS I aimed my experiments mostly towards verifying the properties of this problem that we discussed in class rather than answering questions about the quality. I did however

make some modifications to the code such that I could vary the step size parameters μ and β and plot their influence and the convergence speed (PCG iterations and overall iterations). These experiments gave me relatively *boring* results as the algorithms seemed particularly robust against bad choices for the μ and β parameters, which in itself is interesting.

For the main focus I chose a sparse $x_0 \in \{-1, 0, 1\}^n$ (see figure 1 left) and a random, zero-mean data matrix. The system is with $55 = m < n = 60$ slightly under-determined and while adding some Gaussian noise (see figure 1 right) I aimed for a poor LS solution and a good regularized solution.

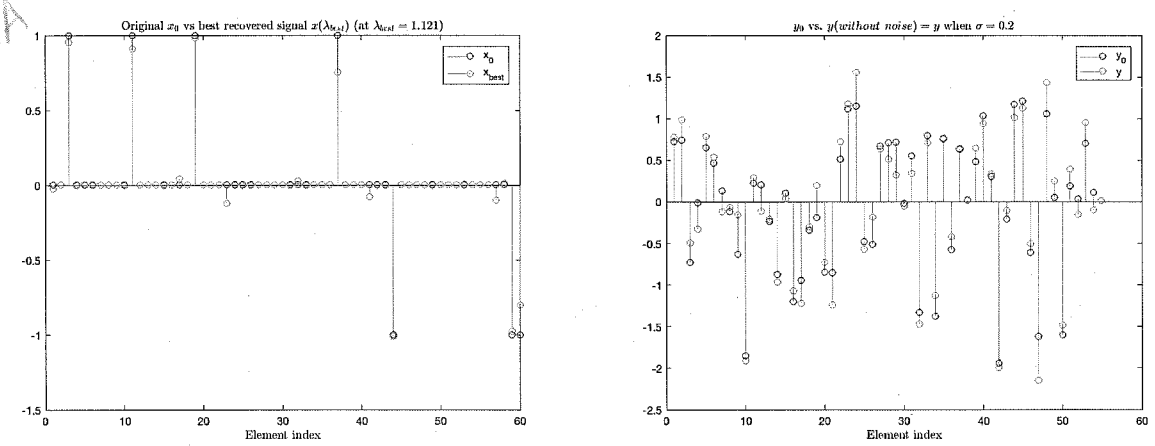


Figure 1: Left: ground truth, x_0, and recovered signal, $x(\lambda_{best})$. Right: Signal, $y = Ax_0$ and signal y_0 after adding Gaussian noise

First, to find the optimal regularization parameter the problem is solved for a wide range of λ's and the mean squared error is minimized to yield $\lambda = \lambda_{best}$ (see figure 2). As the reciprocal of λ is on the horizontal axis the problem converges to the unregularized problem (LS) on the right side of the plot and all x-variables will be 0 on the left side of the plot where $\lambda > ||2A^\top y||_\infty$. The curve for the error clearly has a (local) minimum which is *not* when $\lambda = \infty$. The minimum occurs when $\lambda = \lambda_{best}$ which yields a good match for x (and a perfect match for x after some appropriate thresholding).

The plot of the coordinates of $x(\lambda)$ is shown in figure 2 (right). It shows how the *true* nonzero coordinates (blue) of x *sprout* out of zero first, giving a perfect sparse solution even when $\lambda > \lambda_{best}$. At around λ_{best} even some *false* nonzeros coordinates (red) are becoming nonzeros. Thus, the solution $x(\lambda_{best})$ is slightly inferior in the sense that it does not have the same sparsity as x_0 (see figure 1 right).

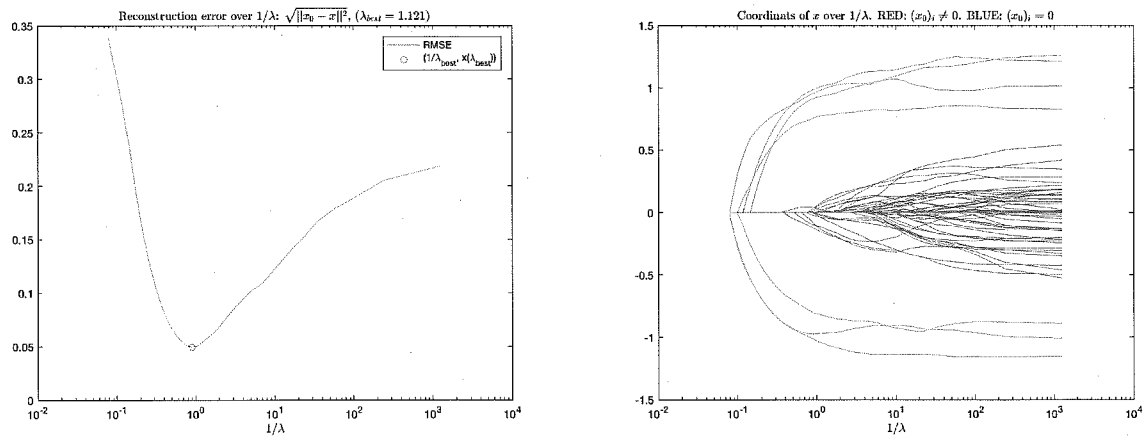


Figure 2: Left: Mean squared error for a range of λ values. Right: Coordinates of minimizers $x(\lambda)$

10/10

Reading Report of Kim et al.

(ECE 695, Reading Assignment 01, Spring 2018)

Overview of the Pape

The paper addresses the issue of solving ℓ_1 regularized optimization problems for tasks such as sparse signal reconstruction, feature selection and others. For ℓ_1 regularized problems we have the assumption that the desired solution is sparse meaning most of its elements should be zero. The authors focus on a specific method called interior-point method, which is an iterative method involving the calculation of a search direction and a step size for each iteration. While existing interior-point methods prior to this work perform well for smaller linear system, they appear to be unpractical when the linear system is huge containing millions of variables. The main contribution of the paper is then a specialized interior-point method that uses preconditioned conjugate gradients algorithm to approximate the search direction calculated from the Newton system. By doing this, the proposed method is able to handle large-scale ℓ_1 regularized problems.

The paper talked about two problems they are interested in: compressed sensing and linear regression. For the case of compressed sensing, they assume the data we want to recover is sparse in a transform domain. As a result, we would expect a transform matrix inside the ℓ_1 term. And when the transform matrix is invertible, the minimization problem can be converted into a standard least-squared program which is the interest of this paper.

Prior Work

Because of the nondifferentiable nature of ℓ_1 regularized problems, generic methods such as ellipsoid method and subgradient methods can be used while being slow. And since ℓ_1 LSP can also be converted into a convex quadratic problem with linear inequality contraints, it can be solved by convex optimization solvers such as interior point method like MOSEK for small or medium sized problems. Specialized methods that exploits matrix-vector operations of $\boldsymbol{A}$ and $\boldsymbol{A}^T$ are able to handle large scale problems. Homotopy based methods were also proposed for solving the same problem.

Key Ideas of the Paper

The first key idea of this paper is the design of the logarithmic barrier for the bound constraint. By design, when x_i approaches v_i, the input to the log is close to zero which in turn produce a huge cost for the objective function. Since $-log(*)$ is infinity for zero inputs, the we almost never expect the solution to be at the barrier.

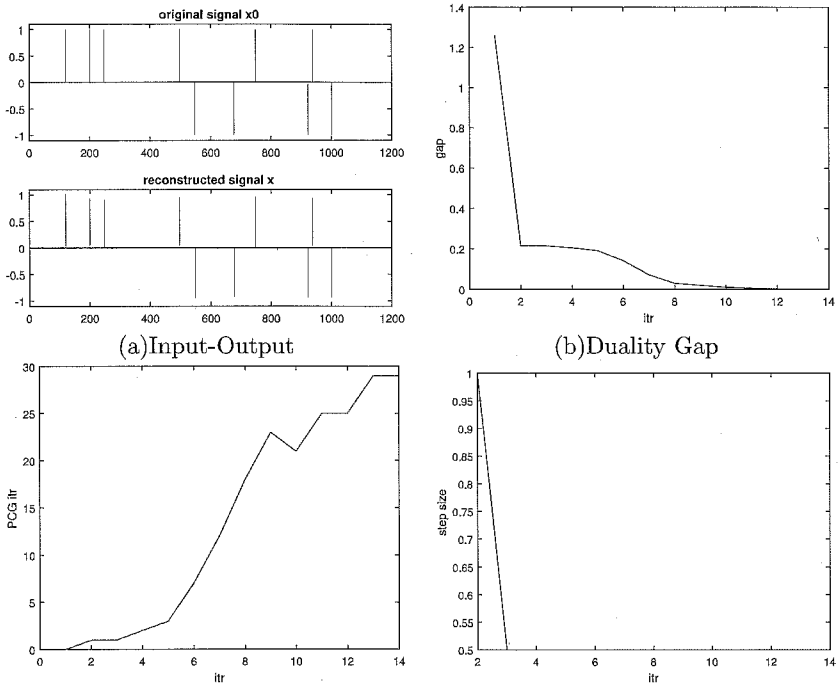
The second key idea is using the PCG method to approximate the search direction resulted from the Newton system. While the approximate solution will never be ideal, it offers a good trade-off between the error and speed which is essential for large-scale problems. And since the PCG solver is iterative for approximation of the solution of the Newton system, the overall method is a Truncated Newton method.

Another key point is the adoption of the following update rule for the parameter t.

$$t = \begin{cases} max\{\mu min\{2n/\eta, t\}, t\} & s \geq s_{min} \\ t, & s < s_{min} \end{cases}$$

s is basically used as a measurement for the proximity of the current solution to the central path. When s is large, the update rule will attempt to increase t so that the solution will be less sub-optimal. The amount of increase for t is determined by $2n/\eta$. This is also reasonable since when the duality gap is large meaning a early iteration of the algorithm, it might not be reasonable to increase t too much.

A fourth key point is the truncation rule which depends on the duality gap. Using this design, in early iterations the duality gap is large and the PCG result will have a lower accuracy, and in later iterations the duality gap is small thus a PCG result with high accuracy is preferred.



(a)Input-Output (b)Duality Gap

(c)PCG Iterations (d)Step Size

Figure 1: Experiment results for sparse signal recovery case with $\beta = 0.5$.

Comments

I ran their code package for a sparse signal recovery problem with signal dimension being 1024, and number of observation being 128. The solution is reached in 14 iterations. The result is similar to the ground truth.

Apparently, for this case the step size is always greater or equal to 0.5, which makes the algorithm to constantly update t for each iteration. From the figures we can also see the duality gap throughout the iterations are pretty small, which means $2n/\eta$ (used in the update rule) is always greater than t since the algorithm converges fast enough. This means the update rule for this example is basically the trivial $t = u^k t_0$ as used by typical interior point methods as mentioned in the paper. The constant step size through iterations seems to be caused by a parameter $\beta = 0.5$ used for line search. If I change to $\beta = 0.4$, the step sizes are oscilating between 0 and β. This then cause the update rule to only update once every two iterations. This change causes the algorithm to reach a solution after 24 iterations, which is a lot slower then using a $\beta = 0.5$, although the final solution looks identical. This means updating t is crucial for a fast convergence of the algorithm. With a large-scale problem where $2n/\eta$ is basically always larger than t, this means the proposed update rule just simply skips some iteration based on the choice of β, and the skippings apparently does not improve the performance or the convergence speed. So after running the experiments, I'm actually sort of doubtful about their update rule and its usefulness.

Another thing I realize is that when I increase the dimension of the ground truth signal by a factor of 20, the algorithm terminates with a single iteration and output a vector of zeros. Decreasing the value of λ will stop the algorithm producing zeros as the output, while the output itself is not accurate. I suspect this is due to the PCG not performing very well when the ground truth signal is very sparse, to improve the performance of PCG I decrease the relative tolerance used by it hoping the solution of PCG is a better approximation of the Newton system. However, making this change does not produce a more accurate final solution. After second thought, by increasing dimensionality of the signal while not increasing the number of spikes and keeping the noise level unchanged, I'm basically just increasing the noise signal ratio, and the performance drop might just be reasonable.

10/10

Reading Report of S. J Kim, K. Koh, et al.

(ECE 695, Reading Assignment 01, Spring 2018)

Overview of the Paper

This paper focuses on solving the problem y=Ax+v in an efficient way. To solve this problem, the authors use the ℓ_1-regularized least squares programs (LSP) formulation given by:

$$minimize||Ax - y||_2^2 + \lambda||x||_1 \tag{1}$$

Kim et al propose their own optimization method for very large sparse problems based of a combination of different steps such as Interior Point Method , Newton's Truncated Method, and Lagrange Duality.

The authors then demonstrate their method attains accurate solutions in a relatively shorter time than other optimization algorithms, and they show they can generalize their method to more ℓ_1 minimization problems.

In order for this algorithm to work efficiently, they need certain conditions. For example, they need the system to be number of unknowns to be much greater than the number of observations. In addition, to obtain a non-zero solution they need $\lambda < ||2A^T y||_\infty$. Similarly, when this method is applied to compressed sensing, the author assumes there is a linear transform W which can create a sparse representation of signal z in a different domain (paper's equation (6)).

Prior Work

The paper presents a wide range of prior approaches based off ℓ_2 and ℓ_1 optimization algorithms. For example, the author talks about Tikhonov's quadratic method (ℓ_2) to solve the initial problem. This method yields a closed form solution but the matrix inversion is computationally expensive when the dimensions are high. Regarding the the ℓ_1-LSQ approaches, they do not have a close form solution due to the nature of the ℓ_1 norm, so all the algorithms are iterative. The author talks about ellipsoid or sub-gradient methods, but he mentions that these approaches are slow. The author also mentions interior point methods such as MOSEK; however this approach does not work efficiently for large problems. There are conjugate descend methods that are very fast but their results do not obtain high accuracy. The author mentions that Figueiredo et al proposed a "Gradient Projection Method" for sparse inverse problems that does work efficiently for large dataset, so it would be interesting to explore the comparison between these two approaches.

Key Ideas of the Paper

First, the author makes sure that λ is within the desired constraints. This condition will ensure the method yields non-zero solutions. Second, the paper derives a Lagrange Dual Function for equation (1) by letting $z = Ax - y$, and showing there is a dual function $v(z)$ that will help to minimize the primal function.

For the Interior Point Method, equation (1) is expressed as a convex quadratic minimization function with linear constrains (replacing x from the second term of equation (1) with variable u_i), where $-u_i < x < u_i$. These constrains are added as a logarithmic function $\Phi(x, u)$ to the the quadratic constrained equation to create a new function to minimize. Therefore the new equation to be minimized is:

$$\phi_t(x, u) = t||Ax - y||_2^2 + t \sum_{i \in [1,n]} \lambda u_i + \Phi(x, u) \tag{2}$$

The author uses the Truncated Newton Method to minimize the Interior Point Function. The Truncated Newton Method computes first a PCG search direction before applying Newton's Method in order to save computational cost. Finally, at each iteration, the author varies the parameter t from 0 to ∞, using the Dual Function $v^*(t)$ as a reference. The author performs these iterations and updates the value of t until the desired duality gap is achieved. The complete and detailed algorithm is given in the paper.

Figure 1

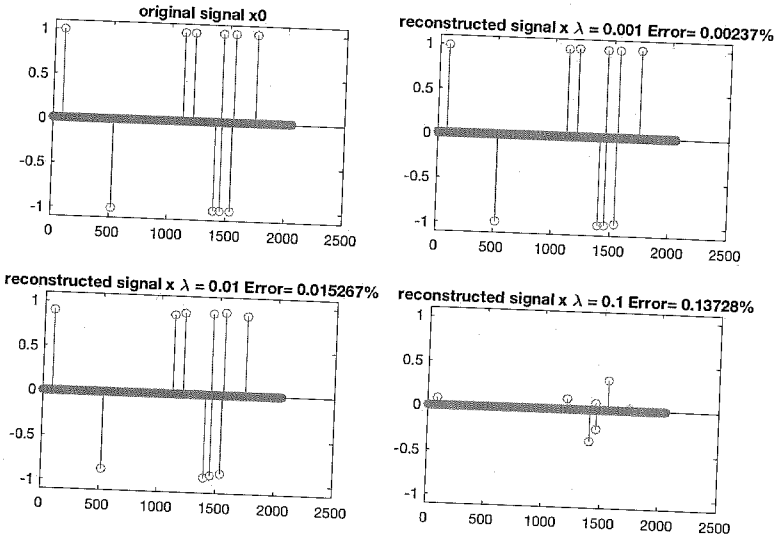


Figure 1: Code Outputs for Different Values of λ

Comments

The algorithm itself seems very fast and accurate. The comparisons shown in the results between TNIPM and other top notch algorithms such as l1-magic, and MOSEK have one order of magnitude of difference for running times with large problems. For this section, I will show results of their sample codes, vary their parameters, and compare with other optimization algorithms.
One possible problem I find is that this method relies on sparsity, or that the signal has a sparse representation in another domain. For example, in a signal with 2048, if it contains only 10 non-zero elements, the method attains a 0% error rate. If the number of non-zero elements is increased to 100, the method achieves 6% error rate. Finally, if the number of non-zero elements is increased to 1000, the algorithm achieves a 51% error rate, which is worse than a random guess. This error could happen due to the regularization parameter pushing as many arguments as possible to be zero. Therefore, both the search direction and Newtons approach could yield non-optimal solutions due to the increase in dimensionality. Similarly, the optimization results can change for different choices of λ within the allowed range. For example, the sample codes provide a DCT transform as a measurement matrix for a random sparse signal with 10 spikes within 2048 variables and 128 measurements. For this case, $\lambda_{max} = 1.6$. When $\lambda \in [0, 0.07]$, the signal can be mostly recuperated; however, for values above 0.08, the recuperated signal becomes random.This issue could influence the optimization for larger problems and the authors mention the optimal value of λ can be found by trial and error, the difficulty of this task can become cumbersome when dealing with a very large problem.
Finally, l1-magic has a library available which performs similar operations to the methods used in this problem such as log-barrier optimization and interior point method optimization; however, due to time,space constrains and code compatibility (shape measurement matrix A in the same way it is created for this problem), the comparison will not be included in this report.

what if you also increase size of signal?

References

[1] S. J. Kim, K. Koh, M.Lustig, Sthepen Boyd, and Dimitry Gorivnesky, "An interior-point method for large-scale l1-regularized least squares," *Journal of Selected Topics in Signal Processing*, vol. 1, no. 4, pp. 606–616, 2007.

10/10

Reading Report of Kim et al.

(ECE 695, Reading Assignment 01, Spring 2018)

Overview of the Pap

In this paper, a low computational complexity with high accuracy algorithm is proposed for solving the following l_1-regularized least square minimization problem

$$\min_x \quad \|Ax - y\|_2^2 + \lambda \|x\|_1 \tag{1}$$

Comparing with existing algorithm, the low computational complexity nature makes the proposed algorithm more favorable for large problems, i.e., problems with large number of variables to be optimized. The authors provide analytical results of performance, and computational complexity of the proposed algorithm. Finally, numerical results are provided for comparison of the proposed algorithm and existing ones.

Prior Work

First, the problem (1) has no closed-form solution, so one must solve it numerically. Since (1) is convex but non-differentiable, algorithms for such problems, including ellipsoid method and sub-gradient method, can be applied to solve (1). These algorithms do not perform well in terms of computational complexity. Another idea of solving (1) comes from the fact that (1) is equivalent to convex quadratic program (QP) with linear constraint. In this way, standard algorithms such as interior-point method can be adopted to solve (1). Note that standard interior-point methods are less suitable for large problems involves matrix-vector operations with A and A^T, which appears in solving (1). There are other algorithms proposed before this paper to solve (1), such as coordinate-wise descent methods, a fixed-point continuation method, Bregman iterative regularization based methods, sequential subspace optimization methods, bound optimization methods, iterated shrinkage methods, gradient methods, and gradient projection algorithms. Note that some of the algorithms above performs well in terms of computational complexity for large problems.

Key Ideas of the Paper

In this paper, instead of working on (1) directly, the authors tackle its equivalent convex QP (with linear constraint). Since interior-point methods are standard algorithm for solving convex QP, and they suffer from (relatively) high computational complexity when dealing with QPs which are transformed from (1) of large size, a natural idea is to develop a low computational complexity version of interior-point method for such problem. Because the high computational complexity comes from finding the exact search direction, the author propose to adopt the preconditioned conjugate gradients algorithm to compute the search direction, in order to reduce the computational complexity. In a nutshell, the authors propose *an interior-point method with the preconditioned conjugate gradients algorithm for computing the search direction* to solve the equivalent problem of (1).

Comments

I think this paper deserves high citations and the best paper award, since it solve a practical and widely applicable problem, i.e., the computational complexity issue of solving (1) of large size. Specifically, a detailed description of the proposed algorithm is provided, including the (brief) theory behind it, each step of the algorithm, and the guideline of parameter selection in the algorithm, which makes researchers and engineers easy to implement (without downloading the code from the authors website), customize (according to the parameter selection guideline), and possibly improve (according to theory provided) the propose algorithm. However, there are still some parts which can be further improved, in my perspective, which will be list in the following.

First, the authors do not provide convergence analysis of the proposed algorithm. In the paper, the authors state that convergence of *interior-point method with backtracking line search and exact search directions* is guaranteed. However, the proposed algorithm adopts *inexact search directions*, and the authors

like reading a paper. interesting

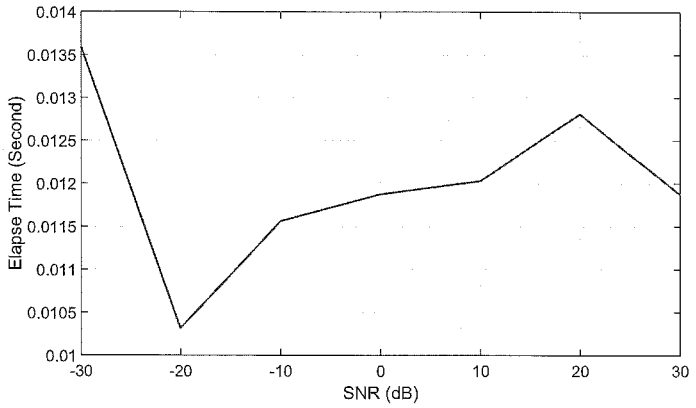


Figure 1: Performance of proposed algorithm for different SNR.

do not provided analytical performance guarantee of such algorithm. (In this paper, the authors only states that *the update rule* **appears** *to be quite robust and work well when combined with the PCG algorithm we will describe soon.*) In other words, according to the analysis in this paper, it is possible that the algorithm never converges in some cases.

Second, the authors give the computational complexity for some parts (in terms of flops) of the algorithm, e.g., the computational complexity of some matrix-vector products, but they do not provide the computational complexity of *the whole algorithm*, i.e., given the parameters such as problem size and relative tolerance, what is the computational complexity of the proposed algorithm. Such analysis is important due to the following reasons: first, for a system designer, it is necessary to design the system parameters. For example, a magnetic resonance imaging (MRI) machine designer needs the computational complexity to have the knowledge the answer of the following questions (for a given hardware capability): for a given resolution and accuracy, how much time is needed for the MRI machine to finish the task, what is the best result (in terms of resolution and accuracy) given the maximum tolerance waiting time, etc. For research purposes, it is also important to know the analytical computational complexity for performance comparison. In this paper, numerical results are provided to show the performance of the proposed algorithm as well as other algorithms. However, without analytically proof, even if 100 examples show that the proposed algorithm works better than other algorithms, we still cannot assert that the proposed algorithm outperforms other algorithm for the 101 example.

Third, for the numerical results, it would be better to provide some results when the elements of matrix A are correlated, which often happen in practical examples. The scores of GRE verbal and TOEFL should be correlated is an example mentioned in class. Since performance of an algorithm could be quite different for A with different properties, it would be better to show that results. Same story for the noise.

By the way, since I wonder the effect of signal to noise ratio (SNR) on the performance, I ran a simulation to investigate such effects. The parameters I adopted is as follows: $\lambda = 0.01, \epsilon_{\text{rel}} = 0.01$,

$$A = \begin{bmatrix} 1 & 0 & 0 & 0.5 \\ 0 & 1 & 0.2 & 0.3 \\ 0 & 0.1 & 1 & 0.2 \end{bmatrix}, x_0 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix},$$

noise is generated according to the Gaussian model $\mathcal{N}(0, \sigma^2 \mathbf{I})$. The performance metric I adopted is the CPU time of the algorithm, with the result being averaged over 100 noise realizations. Figure show the results. It is shown that the performance is affected by SNR of signal, in the example adopted -20 dB SNR yields the best performance. However, the difference is not significant in our simulation, possibly due to the small problem size. It is interesting to see the performance of large size problems.