

Midterm report

Ruicheng Ao 1900012179

February 16, 2022

1 ℓ_1 minimization

1.1 Problem settings

In this section, we consider the following optimization problem

$$\min_x \quad \mu \|x\|_1 + \|Ax - b\|_\infty, \quad (1)$$

where $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$ are given.

Problem (1) can be reformulated into a constrained problem by introducing constraint $y = Ax - b$:

$$\begin{aligned} \min_{x,y} \quad & \mu \|x\|_1 + \|y\|_\infty \\ \text{s.t.} \quad & y = Ax - b \end{aligned} \quad (2)$$

with its dual problem given by

$$\begin{aligned} \max_\lambda \quad & -b^\top \lambda \\ \text{s.t.} \quad & \|\lambda\|_1 \leq 1, \|A^\top \lambda\|_\infty \leq 1 \end{aligned} \quad (3)$$

1.2 Augmented Lagrangian method

In the past, penalty function methods had been prevailed by directly adding a constraint violation penalty to the target function [2, section 10] and converting the constrained problem into an unconstrained problem. For example, the quadratic penalty method is welcomed due to its simple form and easy implementation. However, these kinds of methods come up with disadvantages such as generating ill-conditioned problems when the penalty factors are large, which impairs the convergence.

The augmented Lagrangian method, however, relieves such difficulty by introducing the Lagrangian multipliers combined with quadratic penalty function. To speak more specifically, by applying such technique, an augmented Lagrangian function of primal problem (1) is of the form

$$L(x, y, \lambda, \tau) = \mu \|x\|_1 + \|y\|_\infty + \lambda^\top (Ax - b - y) + \frac{\tau}{2} \|Ax - b - y\|_2^2. \quad (4)$$

One of the most obvious advantage of this kind of method is implied in that, throughout the optimization process, there is no necessity for the augmented Lagrangian method to set gradually τ to infinity in order to get the optimal point, which avoids ill-conditioned subproblems. Algorithm 1 shows a general framework of augmented Lagrangian method.

Algorithm 1 Augmented Lagrangian method

```

1: Initialize  $x^0, y^0, \lambda^0$ 
2: while Not converge do
3:   Solve  $(x^{k+1}, y^{k+1}) = \arg \min_{x,y} L(x, y, \lambda^k, \tau)$ 
4:   Update Lagrangian multiplier  $\lambda^{k+1} = \lambda^k + \tau(Ax^{k+1} - b - y^{k+1})$ 
5:    $k = k + 1$ 
6: end while
7: return  $x^k, y^k$ 

```

It should be mentioned that, in our form of constraint $y = Ax - b$, one can show that the augmented Lagrangian method is equivalent to the well-known Bregman iterative method (Algorithm 2) when $\tau = 1$ in ALM. [6] has proved the following results of convergence for Bregman iterative method.

Theorem 1. (Yin)

- (a) Suppose the iterate (x^k, y^k) in Algorithm 2 satisfies $Ax^k - y^k = b$, then (x^k, y^k) are the solutions of primal problem (2).
- (b) There exists a number $K < \infty$, such that for any $k > K$, (x^k, y^k) is a solution of (2).

Algorithm 2 Bregman iterative method

```

1: Initialize  $x^0, y^0, f^0$ 
2: while Not converge do
3:   Update  $f^{k+1} = f^k - (Ax^k - y - b)$ 
4:   Solve  $(x^{k+1}, y^{k+1}) = \arg \min_{x,y} H(x, y, f, \tau) = \mu\|x\|_1 + \|y\|_\infty + \frac{\tau}{2}\|Ax - y - f^k\|_2^2$ 
5:    $k = k + 1$ 
6: end while
7: return  $x^k, y^k$ 

```

We apply the fast iterative shrinkage-thresholding algorithm (FISTA) [1] for solving the subproblem in Algorithm 1, which is illustrated in Algorithm 3. We split the augmented Lagrangian function into two parts in order to use FISTA, i.e., $f(x, y) = \lambda^\top (Ax - b - y) + \frac{\tau}{2}\|Ax - b - y\|_2^2$, $g(x, y) = \mu\|x\|_1 + \|y\|_\infty$. The proximal operator of $tg(x, y)$, $t > 0$ composes of two parts. One is $\text{prox}_{t\|\cdot\|_1}(x)$, which can be expressed explicitly as

$$\text{prox}_{t\|\cdot\|_1}(x)_i = \begin{cases} x_i + t, & x_i < -t, \\ 0, & |x_i| \leq t, \\ x_i - t, & x_i > t, \end{cases} \quad (5)$$

The other part $\text{prox}_{t\|\cdot\|_\infty}(y)$, however, is more difficult to compute directly. Nevertheless, it can be obtained by applying Moreau decomposition [4], i.e.,

$$y = \text{prox}_h(y) + \text{prox}_{h^*}(y), \quad (6)$$

where h is any convex function and $h^*(y) = \sup_z (z^\top y - h(z))$. Note that the conjugate of $\|\cdot\|_\infty$ has the form of $\sup_z (z^\top y - t\|z\|_\infty) = \begin{cases} 0, & \|y\|_1 \leq t, \\ +\infty, & \|y\|_1 > t, \end{cases}$ we have $\text{prox}_{t\|\cdot\|_\infty}(y) = y - \text{prox}_{1\|x\|_1 \leq t}(y) = y - \text{proj}_{\|\cdot\|_1 \leq t}(y)$. The projection

of y to l_1 ball of radius t can be obtained by solving the following equations

$$\text{proj}(y)_i = \text{sign}(y_i)(|y_i| - \lambda)_+ = \begin{cases} 0, & |y_i| \leq \lambda, \\ \text{sign}(y_i)(|y_i| - \lambda), & |y_i| > \lambda, \end{cases} \quad (7)$$

where $\lambda \geq 0$ is the minimal nonnegative number satisfying $\sum_{i=1}^m (|y_i| - \lambda)_+ \leq t$. This subproblem can be solved using bisection method in time of $\mathcal{O}(m \log m)$.

The following theorem in [1] demonstrates the general convergence of FISTA:

Theorem 2. *Let $\{x^k\}$ be the iterates of FISTA, then for any $k \geq 1$, $F(x) = f(x) + g(x)$ satisfies*

$$F(x^k) - F(x^*) \leq \frac{2L\|x^0 - x^*\|_2^2}{(k+1)^2}, \forall x^* \text{ optimal.} \quad (8)$$

Thus the computation complexity for finding an ϵ -optimal point is $\mathcal{O}(\frac{nm \log mL^{\frac{1}{2}}}{\epsilon^{\frac{1}{2}}})$ with the cost per iteration $\mathcal{O}(nm \log m)$.

Algorithm 3 Fast iterative shrinkage-thresholding algorithm

- 1: Given f, g , initial point x^0 , Lipschitz coefficient L of ∇f
 - 2: Set $y^1 = x^1, t^1 = 1$
 - 3: **while** Not converge **do**
 - 4: Solve $x^k = \arg \min_x \text{prox}_{g(x)/L}(y^k - \frac{1}{L} \nabla f(y^k))$
 - 5: Update $t^{k+1} = \frac{1 + \sqrt{1 + 4(t^k)^2}}{2}$
 - 6: Update $y^{k+1} = x^k + \frac{t^k - 1}{t^k + 1}(x^k - x^{k-1})$
 - 7: $k = k + 1$
 - 8: **end while**
 - 9: **return** x^k
-

To the end of this section, we illustrate the practical implementation of augmented Lagrangian method in Algorithm 4.

Algorithm 4 Augmented Lagrangian method

- 1: Given tolerance tol, ϵ^0 , penalty parameter τ^0 , initial point $(x^0, y^0), \lambda^0$
 - 2: **while** $|\mu\|_1 + \|y\|_\infty + b^\top \lambda + \|Ax - b - y\|_2^2 > tol$ **do**
 - 3: Using FISTA to obtain ϵ^k -optimal solution (x^k, y^k) of $L(x, y, \lambda^k, \tau^k)$
 - 4: Update Lagrangian multiplier $\lambda^{k+1} = \lambda^k + \tau^k(Ax^k - b - y^k)$
 - 5: Choose initial point for FISTA of the next subproblem and update tolerance $\epsilon^{k+1} \leq \epsilon^k$, penalty parameter $\tau^{k+1} \geq \tau^k$.
 - 6: $k = k + 1$
 - 7: **end while**
 - 8: **return** x^k, y^k
-

1.3 Alternating direction method of multipliers

In this section, we demonstrate alternating direction method of multipliers (ADMM) for solving primal problem (2). We rewrite the augmented Lagrangian function as follows

$$L(x, y, \lambda, \tau) = \mu\|x\|_1 + \|y\|_\infty + \lambda^\top (Ax - b - y) + \frac{\tau}{2}\|Ax - b - y\|_2^2. \quad (9)$$

In each iteration of ALM, (x^k, y^k) are updated simultaneously, which leads to costly computation. To address this problem, ADMM modifies the iteration by updating x^k, y^k, λ^k separately. We demonstrate the framework of ADMM in Algorithm 5. The convergence of ADMM is established in [3].

Algorithm 5 Alternating direction method of multipliers

- 1: Initialize $x^0, y^0, \lambda^0, \gamma \in (0, \frac{\sqrt{5}+1}{2})$
 - 2: **while** Not converge **do**
 - 3: Solve $x^{k+1} = \arg \min_x L(x, y^k, \lambda^k, \tau)$
 - 4: Solve $y^{k+1} = \arg \min_y L(x^{k+1}, y, \lambda^k, \tau)$
 - 5: Update Lagrangian multiplier $\lambda^{k+1} = \lambda^k + \gamma\tau(Ax^{k+1} - b - y^{k+1})$
 - 6: **end while**
 - 7: **return** x^k, y^k
-

However, in some situations, solving the subproblem $x^{k+1} = \arg \min_x L(x, y^k, \lambda^k, \tau)$ is intractable. [5] proposed a linearized ADMM scheme demonstrated in Algorithm 6. Each iteration can be proceed quickly by applying proximal operators as mentioned before. Also, since there is no difficulty to directly compute y^{k+1} with the help of proximal operators, we can update y^k by just solving the subproblem. The following theorem in [5] establishes the convergence of linearized ADMM.

Theorem 3. Suppose the stepsize α, γ satisfy $\alpha\lambda_{\max} + \gamma < 2$, where λ_{\max} denotes the largest eigenvalue of $A^\top A$. Then for primal problem $\mu\|x\|_1 + \|y\|_2^2$, the iterates generated by linearized ADMM converges globally to the optimal points.

Note that the convergence analysis is applied to quadratic function of y , nevertheless, it does not hamper the performance of such method. Since the iterates always converge fast to the optimum, where $y = 0$.

Algorithm 6 Linearized ADMM

- 1: Initialize $x^0, y^0, \lambda^0, \gamma \in (0, \frac{\sqrt{5}+1}{2})$, stepsize α
 - 2: **while** Not converge **do**
 - 3: Solve $x^{k+1} = \arg \min_x \mu\|x\|_1 + \langle A^\top (\lambda + \tau(Ax^k - b - y^k)), x \rangle + \frac{1}{2\alpha}\|x - x^k\|_2^2$
 - 4: Solve $y^{k+1} = \arg \min_y L(x^{k+1}, y, \lambda^k, \tau)$
 - 5: Update Lagrangian multiplier $\lambda^{k+1} = \lambda^k + \gamma\tau(Ax^{k+1} - b - y^{k+1})$
 - 6: **end while**
 - 7: **return** x^k, y^k
-

1.4 Numerical results

We generate data for experiments through MATLAB code provided in project requirement, i.e.,

```
A = randn (m,n);
u = sprandn (n,1,0.1);
b = A * u;
mu = 1e - 2
```

The random seed is fixed for so that the experiments can be repeated. $(m, n) = (256, 128), (512, 256), (1024, 512), (2048, 1024)$ are chosen. We compare the performance of Mosek, Gurobi on CVX, augmented Lagrangian method and linearized ADMM on the primal problem. ALM is implemented with both FISTA/ ISTA for comparison. For parameters, we choose $\epsilon^k = 1e - 8, \tau = 1e - 3$ and $\gamma = \frac{\sqrt{5}+1}{2}, \alpha = \frac{1}{\tau\lambda_{\max}}$ for ADMM. We use randomly generated initial points. The stopping toleration is chosen to be $1e - 10$ based on the difference of two following steps.

The numerical results are displayed in Table 1 , we do not list the number of iterations since the time shows much more importance. From the table, we see that the ALM and linearized ADMM outperform the CVX toolbox. High-quality solutions are obtained with negligible violataion of constraints. For ALM, the superiority of FISTA can be found in the results. In all scales of problems, linearized ADMM shows the fastest convergence speed, which partly derives from its exploitation of the problem structure and less proximate operators used.

Algorithm	$(m, n) = (128, 256)$					
	time	optvalue	violation	optgap	error to mosek	$\ x\ _1$
Mosek	9.60e-01	1.76e-07	-	-	-	2.33e+01
Gurobi	1.06e+00	2.33e-09	-	-	3.74e-08	2.33e+01
ALM with ISTA	4.18e-02	4.74e-09	7.75e-11	1.51e-11	3.65e-08	2.33e+01
ALM with FISTA	2.34e-02	3.51e-09	5.64e-11	2.00e-11	3.65e-08	2.33e+01
ADMM	7.25e-03	2.01e-10	1.68e-10	3.04e-13	3.65e-08	2.33e+01
Algorithm	$(m, n) = (256, 512)$					
	time	optvalue	violation	optgap	error to mosek	$\ x\ _1$
Mosek	1.12e+00	3.10e-10	-	-	-	4.39e+01
Gurobi	1.47e+00	7.53e-09	-	-	6.32e-10	4.39e+01
ALM with ISTA	2.51e-01	2.23e-09	4.80e-11	3.45e-11	9.81e-11	4.39e+01
ALM with FISTA	7.61e-02	2.96e-09	6.12e-11	3.18e-11	1.18e-10	4.39e+01
ADMM	2.00e-02	6.09e-11	1.42e-10	1.32e-13	4.98e-11	4.39e+01
Algorithm	$(m, n) = (512, 1024)$					
	time	optvalue	violation	optgap	error to mosek	$\ x\ _1$
Mosek	2.27e+00	5.19e-08	-	-	-	8.25e+01
Gurobi	4.29e+00	2.46e-09	-	-	1.99e-08	8.25e+01
ALM with ISTA	2.81e+00	3.77e-09	8.25e-11	4.17e-11	6.31e-09	8.25e+01
ALM with FISTA	3.60e-01	3.76e-09	8.14e-11	2.41e-12	6.41e-09	8.25e+01
ADMM	8.84e-02	1.80e-10	1.07e-09	4.65e-12	6.35e-09	8.25e+01
Algorithm	$(m, n) = (1024, 2048)$					
	time	optvalue	violation	optgap	error to mosek	$\ x\ _1$
Mosek	7.50e+00	1.00e-08	-	-	-	1.69e+02
Gurobi	5.76e+01	7.11e-07	-	-	1.06e-07	1.69e+02
ALM with ISTA	1.28e+02	1.29e-09	4.48e-11	7.04e-12	1.24e-08	1.69e+02
ALM with FISTA	5.84e+00	2.42e-09	6.89e-11	1.48e-11	1.24e-08	1.69e+02
ADMM	1.36e+00	8.03e-12	1.36e-10	7.61e-15	1.24e-08	1.69e+02

Table 1: Numerical results for L_1 minimization. Violation means relative constraint violation defined by $\frac{\|Ax-y-b\|_2}{\|b\|_2}$. CPU time is recorded. Optvaule represents $\frac{|f(x,y)-f(x^*,y^*)|}{|f(x^*,y^*)|}$, where $f(x,y) = \mu\|x\|_1 + \|y\|_\infty$. Error to mosek denotes the relative l_2 difference with the solution of mosek.

References

- [1] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.
- [2] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.
- [3] Roland Glowinski and Patrick Le Tallec. *Augmented Lagrangian and operator-splitting methods in nonlinear mechanics*. SIAM, 1989.
- [4] JB Hiriart-Urruty and Ph Plazenet. Moreau’s decomposition theorem revisited. In *Annales de l’IHP Analyse non linéaire*, volume 6, pages 325–338, 1989.

-
- [5] Junfeng Yang and Yin Zhang. Alternating direction algorithms for ℓ_1 -problems in compressive sensing. *SIAM journal on scientific computing*, 33(1):250–278, 2011.
- [6] Wotao Yin, Stanley Osher, Donald Goldfarb, and Jerome Darbon. Bregman iterative algorithms for ℓ_1 -minimization with applications to compressed sensing. *SIAM Journal on Imaging sciences*, 1(1):143–168, 2008.