A Brief Review of Alternating Direction Method Of Multipliers (ADMM)

Abhishek Aich
aaich@ece.ucr.edu

This report is a brief overview of the paper [1]. Note that, vectors and matrices are denoted in bold lowercase and bold uppercase, respectively.

1 Introduction

The Alternating Direction Method of Multipliers (ADMM) is an algorithm developed for distributed convex optimization. It takes the form of divide and conquer approach in which, the solutions to local sub-problems are coordinated to find a solution to the global problem. ADMM is said to have the advantages of a dual decomposition and the augmented Lagrangian methods for constrained optimization. This algorithm is closely related to many other well known algorithms in literature such as Dykstra's alternating projections method and Bregman iterative algorithms for ℓ_1 optimization problems in signal processing.

2 Preliminaries

The following optimization algorithms set the path for ADMM and hence, a review of these will help build the intuition for the same. We will look at the concept of Dual Ascent, Dual decomposition, Augmented Lagrangians and the Method of Multipliers and, then move on to ADMM followed by a simple application of this algorithm.

2.1 Dual Ascent

Let us consider the following equality-constrained convex optimization problem:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{Ax} = \mathbf{b}. \end{aligned} \tag{2.1}$$

Here, $\mathbf{A} \in \mathbb{R}^{m\times n}, \mathbf{b} \in \mathbb{R}^m$, and $f : \mathbb{R}^n \to \mathbb{R}$ is a convex function. With $\mathbf{y}$ as the Lagrange multiplier, the Lagrangian of (2.1) gives

$$L(\mathbf{x}, \mathbf{y}) = f(\mathbf{x}) + \mathbf{y}^T(\mathbf{Ax} - \mathbf{b}). \tag{2.2}$$

We can then write the dual function of (2.1) as follows:

$$\begin{aligned} g(\mathbf{y}) &= \inf_{\mathbf{x}\in\mathbb{R}^n} L(\mathbf{x}, \mathbf{y}) \\ &= -f^*(-\mathbf{A}^T\mathbf{y}) - \mathbf{b}^T\mathbf{y}, \end{aligned} \tag{2.3}$$

where f^* is the convex conjugate of f. So, we have to optimize the dual function as.

$$\max_{\mathbf{y}\in\mathbb{R}^n} \; g(\mathbf{y}) \tag{2.4}$$

Assuming that strong duality holds, the solution of the primal and dual problems are said to be same. The primal optimal solution $\mathbf{x}^*$ can be recovered from the dual optimal solution $\mathbf{y}^*$ as follows:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathbb{R}^n} \; L(\mathbf{x}, \mathbf{y}^*) \tag{2.5}$$

provided this minimizer is unique. Coming to the *Dual Ascent* method, we solve (2.3) using gradient descent method. Assuming that g is differentiable, this is done in the following way:

1. Find $\mathbf{x}^+ = \arg\min_{\mathbf{x}\in\mathbb{R}^n} \; L(\mathbf{x}, \mathbf{y})$

2. Compute the residual for the equality constraint,*i.e.* $\nabla g(\mathbf{y}) = \mathbf{Ax}^+ - \mathbf{b}$.

The iteration updates in this method are

$$\begin{aligned} \mathbf{x}^{k+1} &\triangleq \arg\min_{\mathbf{x}\in\mathbb{R}^n} \; L(\mathbf{x}, \mathbf{y}^k) && \text{(minimization step)} \\ \mathbf{y}^{k+1} &\triangleq \mathbf{y}^k + \alpha^k(\mathbf{Ax}^{k+1} - \mathbf{b}) && \text{(for dual variable update)} \end{aligned} \tag{2.6}$$

where α^k is the step size. Finally, this algorithm is called 'dual ascent' since the dual function in (2.3) increases in each step ($g(\mathbf{y}^{k+1}) > g(\mathbf{y}^k)$) with appropriate choice of α^k.

2.2 Dual Decomposition

Now, let's further breakdown the objective function $f(\mathbf{x})$ as

$$f(\mathbf{x}) = \sum_{i=1}^{N} f_i(\mathbf{x}_i), \tag{2.7}$$

where $\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N]$ and each $\mathbf{x}_i \in \mathbb{R}^{n_i}$ is a sub-vector of $\mathbf{x}$. Similarly, partitioning $\mathbf{A} = [\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_N]$, we get

$$\mathbf{A}\mathbf{x} = \sum_{i=1}^{N} \mathbf{A}_i \mathbf{x}_i. \tag{2.8}$$

The Lagrangian then, can be written as

$$\begin{aligned} L(\mathbf{x}, \mathbf{y}) &= \sum_{i=1}^{N} L_i(\mathbf{x}_i, \mathbf{y}) \\ &= \sum_{i=1}^{N} \left(f_i(\mathbf{x}) + \mathbf{y}^T \mathbf{A}_i \mathbf{x}_i - \frac{1}{N} \mathbf{y}^T \mathbf{b} \right) \end{aligned} \tag{2.9}$$

Clearly, the $\mathbf{x}$-minimization step in (2.6) has now been split into N separate problems that can be solved parallely. Hence, the update steps are

$$\begin{aligned} \mathbf{x}^{k+1} &\triangleq \arg \min_{\mathbf{x}_i \in \mathbb{R}^{n_i}} \; L_i(\mathbf{x}_i, \mathbf{y}^k) & \text{(minimization step)} \\ \mathbf{y}^{k+1} &\triangleq \mathbf{y}^k + \alpha^k (\mathbf{A}\mathbf{x}^{k+1} - \mathbf{b}) & \text{(for dual variable update)} \end{aligned} \tag{2.10}$$

The minimization step in (2.10) is solved in parallel for each $i = 1, 2, \cdots, N$. Consequently, this decomposition in the dual ascent method is referred as the *dual decomposition.*

2.3 Augmented Lagrangians and Method of Multipliers

The convergence of dual ascent algorithm is based on assumptions such as strict convexity or finiteness of f. To avoid such assumptions and ensure robustness to the dual ascent algorithm, Augmented Lagrangian methods were developed. The *augmented* Lagrangian for (2.1) is given by

$$L_\rho(\mathbf{x}, \mathbf{y}) = f(\mathbf{x}) + \mathbf{y}^T (\mathbf{A}\mathbf{x} - \mathbf{b}) + \frac{\rho}{2} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 \tag{2.11}$$

where $\rho > 0$ is called the penalty parameter. This augmented Lagrangian in (2.11) can be viewed as the unaugmented Lagrangian of the problem

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & f(\mathbf{x}) + \frac{\rho}{2}\|\mathbf{A}\mathbf{x}-\mathbf{b}\|_2^2 \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \end{aligned} \tag{2.12}$$

(2.12) is equivalent to (2.1) because for any feasible $\mathbf{x}$, the term $\frac{\rho}{2}\|\mathbf{A}\mathbf{x}-\mathbf{b}\|_2^2$ becomes zero. The associated dual function of (2.12) is

$$g_\rho(\mathbf{y}) = \inf_{\mathbf{x}\in\mathbb{R}^n} \; L_\rho(\mathbf{x},\mathbf{y}) \tag{2.13}$$

Adding the term $\frac{\rho}{2}\|\mathbf{A}\mathbf{x}-\mathbf{b}\|_2^2$ to $f(\mathbf{x})$ makes $g_\rho(\mathbf{y})$ differentiable under milder conditions than on the original problem.

Next, the gradient of the augmented dual function is found similarly as above. Applying the dual ascent algorithm to the modified problem in (2.12), gives

$$\left.\begin{aligned} \mathbf{x}^{k+1} &\triangleq \arg\min_{\mathbf{x}\in\mathbb{R}^n} \; L_\rho(\mathbf{x},\mathbf{y}^k) \\ \mathbf{y}^{k+1} &\triangleq \mathbf{y}^k + \rho(\mathbf{A}\mathbf{x}^{k+1}-\mathbf{b}) \end{aligned}\right\}\text{Method of Multipliers} \tag{2.14}$$

This method is same as the standard dual ascent, with the penalty parameter ρ is used in place of step size α_k. The advantage of the method of multipliers is that it converges under far more general conditions than dual ascent.

2.3.1 Drawback in Method of Multipliers

While the Method of Mutipliers (MM) improved the convergence properties over the dual ascent method, it is not applicable for all cases. When f is separable, the augmented Lagrangian L_ρ is not separable and consequently the $\mathbf{x}$-minimization step (2.14) cannot be carried out separately in parallel for each $\mathbf{x}_i$. This implies that the basic method of multipliers cannot be used for dual decomposition. This brings us to the main topic of review of this article: Alternating Direction Method of Multipliers.

3 Alternating Direction Method of Multipliers (ADMM)

Alternating Direction Method of Multipliers or ADMM is an algorithm which tries overcome the drawback of MM (as explained in (2.3.1)) by blending the decomposability of dual ascent with the superior convergence properties of MM. ADMM algorithm solves problems in the following form:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n,\mathbf{z}\in\mathbb{R}^m} \quad & f(\mathbf{x}) + g(\mathbf{z}) \\ \text{s.t.} \quad & \mathbf{Ax} + \mathbf{Bz} = \mathbf{c} \end{aligned} \tag{3.1}$$

where $\mathbf{A}\in \mathbb{R}^{p\times n}$, $\mathbf{B}\in \mathbb{R}^{p\times m}$, and $\mathbf{c}\in \mathbb{R}^p$. It is assumed that f and g are convex functions. The main difference between (2.1) and (3.1) is that the vector $\mathbf{x}$ is divided into two parts, $\mathbf{x}$ and $\mathbf{z}$. The optimal value of the problem (3.1) is given by

$$p^* = \inf \left\{ f(\mathbf{x}) + g(\mathbf{z}) \middle| \mathbf{Ax} + \mathbf{Bz} = \mathbf{c} \right\} \tag{3.2}$$

As in MM, form the augmented Lagrangian as

$$\begin{aligned} L\rho(\mathbf{x},\mathbf{y},\mathbf{z}) = f(\mathbf{x}) + g(\mathbf{z}) + \mathbf{z}^T(\mathbf{Ax} + \mathbf{Bz} - \mathbf{c}) \\ + \frac{\rho}{2}\|\mathbf{Ax} + \mathbf{Bz} - \mathbf{c}\|_2^2 \end{aligned} \tag{3.3}$$

ADMM consists of the following iterations

$$\begin{aligned} \mathbf{x}^{k+1} &\triangleq \arg\min_{\mathbf{x}\in\mathbb{R}^n} \quad L_\rho(\mathbf{x},\mathbf{z}^k,\mathbf{y}^k) && \text{(minimization step for } \mathbf{x}) \\ \mathbf{z}^{k+1} &\triangleq \arg\min_{\mathbf{z}\in\mathbb{R}^m} \quad L_\rho(\mathbf{x}^{k+1},\mathbf{z},\mathbf{y}^k) && \text{(minimization step for } \mathbf{z}) \\ \mathbf{y}^{k+1} &\triangleq \mathbf{y}^k + \rho(\mathbf{Ax}^{k+1} + \mathbf{Bz}^{k+1} - \mathbf{b}) && \text{(for dual variable update)} \end{aligned} \tag{3.4}$$

where $\rho > 0$. The MM for (3.1) has the form

$$\begin{aligned} (\mathbf{x}^{k+1},\mathbf{z}^{k+1}) &\triangleq \arg\min_{\mathbf{x}\in\mathbb{R}^n} \quad L_\rho(\mathbf{x},\mathbf{z},\mathbf{y}^k) \\ \mathbf{y}^{k+1} &\triangleq \mathbf{y}^k + \rho(\mathbf{Ax}^{k+1} + \mathbf{Bz}^{k+1} - \mathbf{b}). \end{aligned} \tag{3.5}$$

In (3.5), the augmented Lagrangian is minimized jointly with respect to the two primal variables, while in ADMM (3.4), $\mathbf{x}$ and $\mathbf{z}$ are updated in an alternating manner (hence, the term 'alternating direction') Unlike in MM, separating the minimization over $\mathbf{x}$ and $\mathbf{z}$ into two steps allows for decomposition, when f or g are separable.

References

[1] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine learning*, 3(1):1–122, 2011.