Problem Set (With Solutions): ADMM

1

Performing simple method of multipliers.

1.1

Say we are interested in optimizing the following constrained problem:

Minimize: $f(x)$ under constraint: $Ax = b$

The Lagrangian for this problem can be written as:
$L(x, \lambda) = f(x) + \lambda(Ax - b)$

The optimal solution can therefore be found by iterating through x values, such that:

$x_{k+1} = argmin_x L(x, \lambda_k)$

$\lambda_{k+1} = \lambda_k + \alpha_k(Ax_{k+1} - b)$

1.1.1 Write an expression for the method of multipliers (augmented Lagrangian) that uses a penalty ρ to form a regularization term.

Answer: $L(x, \lambda) = f(x) + \lambda(Ax - b) + \frac{\rho}{2} \parallel Ax - b \parallel_2^2$

1.1.2 Using this new expression, what are we trying to minimize and what is the constraint?

Answer: Minimize: $f(x) + \frac{\rho}{2} \parallel Ax - b \parallel_2^2$ (changed) under constraint: $Ax = b$ (unchanged)

1.1.3 Finally, what are the expressions for the iterations x_{k+1} and λ_{k+1} that can be used to optimize this method of multipliers (augmented Lagrangian) problem?

Answer: $x_{k+1} = argmin_x L(x, \lambda_k)$, $\lambda_{k+1} = \lambda_k - \rho(Ax_{k+1} - b)$

1.2

How does alternating direction method of multipliers (ADMM) differ from the typical method of multipliers?

Answer: For typical method of multipliers, we solve an optimization problem under a given constraint by using an iterative minimization algorithm with a penalty. For an ADMM problem, the objective function of the problem is split into 'sub-problems,' and these smaller problems are solved individually in order to solve the original large-scale problem.

2

Describing the advantages and disadvantages of ADMM and versions of ADMM.

2.1

What is an advantage of decentralized ADMM, when compared to centralized?

Answer: Decentralized ADMM has the potential to operate more quickly than centralized algorithms when a large number of nodes are being used, since centralized algorithms require all nodes to communicate with a central node concurrently. Under decentralized ADMM, nodes only communicate with neighboring nodes, and so this congestion, or ”bottleneck,” can be avoided. In other words, with a high number of nodes, decentralized ADMM can decrease computational complexity and increase speed of operation when compared to centralized.

2.2

What is a disadvantage of decentralized ADMM, when compared to centralized?

Answer: Decentralized algorithms are not well suited for applications with a smaller number of nodes, since coordinating nodes without the use of a central node to control the actions of these nodes can be difficult and computationally expensive.

2.3

What is an advantage of asynchronous ADMM, when compared to synchronous?

Answer: A synchronous algorithm is only as fast as its slowest component. Asynchronous algorithms do not need to ”wait” for slower components, which can reduce waiting time and therefore speed up operation time and overall convergence.

2.4

What is an advantage of distributed algorithms?

Answer: Distributed algorithms are very useful for large data sets, since nodes are able to communicate by using shared or distributed memory, so that data that may be unable to be stored on one computer can be stored using distributed memory.

3

Implementing ADMM for use in the consensus of distributed systems.

3.1

What is the meaning of ”consensus” in regards to problems of distributed systems?

Answer: A consensus problem means that all of the local variables being developed should be equal.

3.2

Suppose we would like to minimize $f(x) = \sum_{i=1} N f_i(x)$, in which our objective function is divided into N parts $f_i(x), i = 1, ..., N$. Also suppose that we implement a global consensus constraint, so that all of the local x_i should be equal. Our problem then becomes:

Minimize $f(x) = \sum_{i=1} N f_i(x)$ under constraint $x_i - z = 0, i = 1, ..., N$

3.2.1 Develop an expression for the augmented Lagrangian $L(x_1, ..., x_n, \lambda)$ for this problem.

Answer: $L(x_1, ..., x_n, \lambda) = \sum_{i=1}^{N} [f(x) + \lambda(x_i - z) + \frac{\rho}{2} \parallel x_i - z \parallel_2^2)]$

3.2.2 What are the resulting ADMM algorithmic expressions for the iterations x_{k+1}, z_{k+1}, and λ_{k+1} that can be used to optimize this problem?

Answer:

$$x_i^{k+1} = argmin_{x_i} L(x_1, ..., x_n, \lambda) = argmin_{x_i}(f(x_i) + \lambda^k(x_i - z^k) + \frac{\rho}{2} \parallel x_i - z^k \parallel_2^2),$$

$$z^{k+1} = \frac{1}{N}\sum_{i=1}^{N}[x_i^{k+1} + (1\rho)\lambda^k]$$

$$\lambda_{k+1} = \lambda_k + \rho(x_{k+1}^i - z^{k+1})$$

3.2.3 How would the ADMM algorithmic expression for the z_{k+1} iterations change if we wanted to use the median of the x_i^{k+1} components, rather than the mean of the x_i^{k+1}, to calculate z_{k+1}?

Answer: Using the median would require sorting the x_i^{k+1} components prior to calculating z_{k+1}, such that x^{k+1} is the sorted vector of x_i^{k+1} values, and then using the expressions:

$$x_{med}^{k+1} = x^{k+1}[\lfloor N/2 \rfloor]$$

$$z^{k+1} = \frac{1}{N}\sum_{i=1}^{N}[x_{med}^{k+1} + (1\rho)\lambda^k]$$