Alternating Direction Method of Multipliers I

Department of Statistics

November 9, 2023

University of Seoul

Dual ascent method

We consider the equality-constrained convex optimization problem

$$\begin{aligned} \text{min} \quad & f(x) \\ \text{subject to} \quad & Ax = b, \end{aligned} \tag{1}$$

where $x \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$ and f is convex function. The Lagrangian is

$$L(x, \nu) = f(x) + \nu^\top (Ax - b).$$

If there exist x^* and ν^* such that $\nabla f(x^*) + A^\top \nu^* = 0$ and $Ax^* = b$ then x^* is the solution of the primal problem by the KKT conditions.

The dual problem is given by

$$\max_{\nu} \inf_{x} L(x, \nu).$$

If the strong duality holds, the optimal x^* satisfies

$$x^* = \mathsf{argmin}_x L(x, \nu^*),$$

where ν^* is the optimal solution of the dual problem. The strong duality means that we can also solve the primal problem through the dual problem.

Since the dual function is always concave, we can maximize the dual function by the gradient ascent method under regularity conditions. The dual ascent method consists of two parts:

- evaluation of the dual function from the Lagrangian function (minimization)

$$g(\nu^k) = \min_x L(x, \nu^k)$$

- computation of the gradient of the dual function and update the dual solution.

$$\nu^{k+1} \quad := \quad \nu^k + \rho \nabla g(\nu^k),$$

where $\rho > 0$ is a learning rate.

Gradient of g

- The dual function $g(\nu^k)$ is computed by $g(\nu^k) = L(x^{k+1}, \nu^k)$, where $x^{k+1} = \text{argmin}_x L(x, \nu^k)$. Note that x^{k+1} is a function of ν^k.
- Let $x^*(\nu) = \text{argmin}_x L(x, \nu)$. Then, $g(\nu) = L(x^*(\nu), \nu)$. The gradient $\nabla g(\nu)$ is given by

$$\begin{aligned}
\nabla g(\nu) &= \frac{\partial L(x^*(\nu), \nu)}{\partial \nu} = \frac{\partial L(x^*, \nu)}{\partial x}\frac{\partial x^*(\nu)}{\partial \nu} + \frac{\partial L(x^*, \nu)}{\partial \nu} \\
&= 0 \times \frac{\partial L(x^*, \nu)}{\partial x} + (Ax - b) \\
&= (Ax - b),
\end{aligned}$$

because x^* is the minimizer of $L(x, \nu)$ for each ν. Thus,

$$\nabla g(\nu^k) = Ax^{k+1} - b$$

Dual Ascent Method for (1)

(1) Set $k = 0$ and $\nu^{(k)}$.

(2) $x^{(k+1)} = \text{argmin}_x \ f(x) + \nu^\top (Ax - b)$

(3) $\nu^{(k+1)} = \nu^{(k)} + \rho(Ax^{(k+1)} - b)$

(4) $k \leftarrow k + 1$ and check the convergence of $x^{(k)}$ and $\nu^{(k)}$.

(5) Repeat (2)-(4) until the solutions are converge.

Example 1 (Dual Decomposition)

Let $f : \mathbb{R}^n \mapsto \mathbb{R}$ and f is separable, that is $f(x) = f_1(x_1) + \cdots + f_k(x_k)$, where $f_i : \mathbb{R}^{n_i} \mapsto \mathbb{R}$, $x_i \in \mathbb{R}^{n_i}$ and $\sum_{i=1}^{k} n_i = n$. Consider the following optimization problem with an equality constraint.

$$\begin{array}{ll} \min & f(x) \\ \text{subject to} & Ax - b = 0. \end{array}$$

Let A_i be a submatrix of A associated x_i. That is, $Ax = A_1x_1 + \cdots + A_kx_k$. Then the Lagrangian is written by

$$L(x, \nu) = L_1(x_1, \nu) + \cdots + L_k(x_k, \nu) + \nu^\top b$$

where $L_i(x_i, \nu) = f_i(x_i) + \nu^\top A_i x_i$.

Then, we can apply x-update to each dual ascent method by x_i.

$$\begin{aligned} x_i^{(k+1)} &:= \operatorname{argmin}_{x_i} L_i(x_i, \nu^{(k)}) \\ \nu^{(k+1)} &:= \nu^{(k)} + \rho(\sum_{i=1}^{n} A_i x_i^{(k+1)} - b) \end{aligned}$$

Method of Multipliers

We consider the convex optimization problem

$$\begin{array}{rl} \text{minimize} & f(x) \\ \text{subject to} & Ax = b \end{array}$$

$x \in \mathbb{R}^p$, $A \in \mathbb{R}^{m \times n}$ and f is convex funciton.

Then, the Lagrangian is

$$L(x, \nu) = f(x) + \nu^\top (Ax - b).$$

An augmented Lagrangian is what gives a 2-norm penalty for equality constraint in the Lagrangian and is defined by

$$L_\rho(x,\nu) = f(x) + \nu^\top (Ax-b) + \frac{\rho}{2} \parallel Ax - b \parallel_2^2$$

where $\rho > 0$.

Applying the dual ascent method to the modified problem is known as the method of multipliers.

$$\begin{aligned} x^{(k+1)} &:= \mathsf{argmin}_x L_\rho(x, \nu^{(k)}) \\ \nu^{(k+1)} &:= \nu^{(k)} + \rho(Ax^{(k+1)} - b) \end{aligned}$$

If f is differentiable, then the optimality conditions are defined by :

$$\begin{aligned} \text{Primal feasibility} \quad &: \quad Ax^* - b = 0 \\ \text{Dual feasibility} \quad &: \quad \nabla f(x^*) + A^\top \nu^* = 0 \end{aligned}$$

where x^*, ν^* is the optimal solution.
And by definition, x^{k+1} minimizes $L_\rho(x, \nu^k)$:

$$\begin{aligned} 0 &= \nabla L_\rho(x^{(k+1)}, \nu^{(k)}) \\ &= \nabla f(x^{(k+1)}) + A^\top(\nu^{(k)} + \rho(Ax^{(k+1)} - b)) \\ &= \nabla f(x^{(k+1)}) + A^\top \nu^{(k+1)} \end{aligned}$$

Thus, our dual update $\nu^{(k+1)}$ makes $(x^{(k+1)}, \nu^{(k+1)})$ dual feasible.

Example 2 (Dual Ascent Method: parallel processing from two data databases)

Denote the empirical risk function defined on database 1 and 2 by f_1 and f_2 and denote the model parameter by x. Considering convex optimization problems

$$\underset{x}{\text{minimize}} \qquad f_1(x) + f_2(x)$$

where $x \in R^p$ and we assume f_1, f_2 are differentiable functions.

(continue the example)

We can reformulate the problem by

$$\begin{array}{rl} \underset{x,z}{\text{minimize}} & f_1(x) + f_2(z) \\ \text{subject to} & x = z. \end{array}$$

$z \in \mathbb{R}^p$ is called an auxiliary variable.

(continue the example)

- Let $L(x, z, \nu) = f_1(x) + f_2(z) + \nu^\top (x - z)$ and set an initialized $\nu^{(k)}$ with $k = 0$.
- Note that $L(x, z, \nu^{(k)})$ splits into two independent functions:

$$L(x, z, \nu^{(k)}) = f_1(x) + \nu^{(k)\top} x + f_2(z) - \nu^{(k)\top} z$$

- Solve the two independent optimization problems on each database.

$$\begin{aligned} x^{(k+1)} &= \underset{x}{\text{argmin}}\ f_1(x) + \nu^{(k)\top} x \\ z^{(k+1)} &= \underset{z}{\text{argmin}}\ f_2(z) - \nu^{(k)\top} z \end{aligned}$$

- Update the dual parameter by

$$\nu^{(k+1)} = \nu^{(k)} + \rho(x^{(k+1)} - z^{(k+1)})$$

Generlization

The general formula, including equality constraint, is

$$\begin{aligned} \text{minimize} \quad & f_1(x) + f_2(z) \\ \text{subject to} \quad & Ax + Bz = c. \end{aligned} \tag{2}$$

The Lagrangian is defined by

$$L(x, z, \nu) = f_1(x) + f_2(z) + \nu^\top (Ax + Bz - c)\,. \tag{3}$$

Since the objective function is separable for x and z, we can apply the maximization with respect to x and z independently in the dual ascent method.

$$\begin{aligned} x^{(k+1)} &:= \operatorname{argmin}_x \; f_1(x) + (A^\top v^{(k)})^\top x \\ z^{(k+1)} &:= \operatorname{argmin}_z \; f_2(z) + (B^\top v^{(k)})^\top z \\ \nu^{(k+1)} &:= \nu^{(k)} + \rho \left(Ax^{(k+1)} + Bz^{(k+1)} - c \right) \end{aligned}$$

Alternating Direction Method of Multipliers [Boyd et al., 2011]

$$\begin{aligned} \min \quad & f(x) + g(z) + \rho\|Ax + Bz - c\|^2 \\ \text{subject to} \quad & Ax + Bz = c, \end{aligned} \tag{4}$$

where $\rho > 0$ The problem (4) is the same solution as (2). Because $Ax + Bz - c = 0$ whenever (x, z) is feasible.

Update rule

$$L_\rho(x, z, \nu) = f(x) + g(z) + \nu^\top (Ax + Bz - c) + \rho\|Ax + Bz - c\|^2$$

- For given $\nu^{(k)}$ and $z^{(k)}$, minimize $L_\rho(x, z^{(k)}, \nu^{(k)})$ w.r.t x.
- For given $\nu^{(k)}$ and $x^{(k+1)}$, minimize $L_\rho(x^{(k+1)}, z, \nu^{(k)})$ w.r.t z.
- For given $x^{(k+1)}$ and $z^{(k+1)}$, update $\nu^{(k+1)} = \rho(Ax^{(k+1)} + Bz^{(k+1)} - c)$.

Investigation of updating rule

- Denote the Lagrangian of (4) by

$$L(x, z, \nu) = f(x) + g(z) + \nu^\top (Ax + Bz - c)$$

- In the view of minimizing (3) $L(x, z, \nu)$ w.r.t (x, z), $L_\rho(x, z, \nu)$ is a majorized function of $L(x, z, \nu)$ at a point on $\{(x, z) : Ax + Bz - c = 0\}$.

Assume f and g are differentiable, then we have two feasibility conditions:

$$\begin{aligned} \text{primal feasibility} \quad &: \quad Ax^* + Bz^* - c = 0 \\ \text{dual feasibility} \quad &: \quad \nabla f(x^*) + A^\top \nu^* = 0 \\ & \quad\;\; \nabla g(z^*) + B^\top \nu^* = 0 \end{aligned}$$

where x^*, z^*, ν^* is the optimal solution.

- primal feasibility: the solution should satisfy the constraint.
- dual feasibility: By the definition of the dual function, the dual variable v^* is feasible if (x^*, z^*) is the minimizer of $f(x) + g(z) + \nu(Ax + Bz - c)$, that is

$$\nabla f(x^*) + A^\top \nu^* = 0 \text{ and } \nabla f(z^*) + B^\top \nu^* = 0$$

Investigation of updating rule: Dual feasibility condition

- Suppose that the dual variable is updated by

$$\nu^{(k+1)} = \nu^{(k)} + \rho(Ax^{(k+1)} + Bz^{(k+1)} - c).$$

- If $z^{(k+1)}$ minimizes $L_\rho(x^{(k+1)}, z, \nu^{(k)})$.

$$\begin{aligned} 0 &= \nabla g(z^{(k+1)}) + B^\top \nu^{(k)} + \rho B^\top (Ax^{(k+1)} + Bz^{(k+1)} - c) \\ &= \nabla g(z^{(k+1)}) + B^\top (\nu^{(k)} + \rho(Ax^{(k+1)} + Bz^{(k+1)} - c) \\ &= \nabla g(z^{(k+1)}) + B^\top \nu^{(k+1)} \end{aligned}$$

Therefore z-update always holds dual feasibility for z. However, the dual feasibility of x is not guaranteed.

Because $x^{(k+1)}$ minimizes $L_\rho(x, z^{(k)}, \nu^{(k)})$,

$$\begin{aligned}
0 &= \nabla f(x^{(k+1)}) + A^\top \nu^{(k)} + \rho(Ax^{(k+1)} + Bz^{(k)} - c) \\
&= \nabla f(x^{(k+1)}) + A^\top (\nu^{(k)} + \rho(Ax^{(k+1)} + Bz^{(k)} - c)) \\
&= \nabla f(x^{(k+1)}) + A^\top (\nu^{(k)} + \rho(Ax^{(k+1)} + Bz^{(k+1)} - c) + \rho(Bz^{(k)} - Bz^{(k+1)})) \\
&= \nabla f(x^{(k+1)}) + A^\top \nu^{(k+1)} + \rho A^\top B(z^{(k)} - z^{(k+1)}).
\end{aligned}$$

Thus, the dual feasibility is written by

$$f(x^{(k+1)}) + A^\top \nu^{(k+1)} = \rho A^\top B(z^{(k+1)} - z^{(k)}) = 0.$$

Stopping criterion

We can define the primal and dual residuals in ADMM at step $k+1$

$$\begin{aligned} \text{Primal residuals} \quad &: \quad r^{(k+1)} = Ax^{(k+1)} + Bz^{(k+1)} - c \\ \text{Dual residuals} \quad &: \quad s^{(k+1)} = \rho A^\top B(z^{(k+1)} - z^{(k)}) \end{aligned}$$

Therefore stopping criterion satisfies that $\|r\|_2$ and $\|s\|_2$ are smaller than any ϵ.

- Primal residuals are defined by primal feasibility.
- Dual residual defined by the first dual optimality conditions.

Alternating Direction Method of Multipliers

- Given x, z, and ν, ρ to some initial value.
- Repeat:
  - $x := \text{argmin}_x L_\rho(x, z, \nu)$
  - $z := \text{argmin}_z L_\rho(x, z, \nu)$
  - $\nu := \nu + \rho(Ax + Bz - c)$
  - Stopping criterion : quit $\|r\| < \epsilon$ and $\|s\| < \epsilon$.

Scaled form of ADMM

Define the residual $r = Ax + Bz - c$; then we have transformed an augmented Lagrangian by

$$\begin{aligned} L_\rho(x, z, \nu) &= f(x) + g(z) + \nu^\top r + \frac{\rho}{2}\|r\|^2 \\ &= f(x) + g(z) + \frac{\rho}{2}\|r + \frac{1}{\rho}\nu\|^2 - \frac{\rho}{2}\|\nu\|^2 \\ &= f(x) + g(z) + \frac{\rho}{2}\|r + u\|^2 + \text{constant}, \end{aligned}$$

where $u = \frac{1}{\rho}\nu$.

The scaled ADMM provides a simpler form of the update formula: let $u^{(k)} = \nu^{(k)}/\rho$.

$$
\begin{aligned}
x^{(k+1)} &:= \operatorname{argmin}_x \left(f(x) + \frac{\rho}{2} \parallel Ax + Bz^{(k)} - c + u^{(k)} \parallel_2^2 \right) \\
z^{(k+1)} &:= \operatorname{argmin}_z \left(g(z) + \frac{\rho}{2} \|Ax^{(k+1)} + Bz - c + u^{(k)}\|^2 \right) \\
u^{(k+1)} &:= u^{(k)} + \left(Ax^{(k+1)} + Bz^{(k+1)} - c \right)
\end{aligned}
$$

Scaled dual ADMM

- Given x, z, and u, ρ to some initial value.
- Repeat:
  - $x :=$ argmin$_x \left(f(x) + \frac{\rho}{2}\|Ax + Bz - c + u\|^2\right)$
  - $z :=$ argmin$_z \left(g(z) + \frac{\rho}{2}\|Ax + Bz - c + u\|^2\right)$
  - $u := u + (Ax + Bz - c)$
  - Stopping criterion : quit $\|r\| < \epsilon$ and $\|s\| < \epsilon$

Using the scaled dual variable, we express the x-update step as

$$\begin{aligned} x^+ &= \underset{x}{\text{argmin}} \left(f(x) + \frac{\rho}{2} \|Ax + Bz - c + u\|_2^2 \right) \\ &= \underset{x}{\text{argmin}} \left(f(x) + \frac{\rho}{2} \|Ax - v\|_2^2 \right), \end{aligned}$$

where $v = -Bz + c - u$ is a known constant vector for the purposes of the x-update. If $A = I$ then

$$x^+ = \underset{x}{\text{argmin}} \left(f(x) + \frac{\rho}{2} \|x - v\|_2^2 \right)$$

Update the z in the same way as x-update.

Definition 3 (Proximal Operator)

For a convex function f

$$\mathsf{prox}_f(v) = \mathsf{argmin}_x f(x) + \frac{1}{2}\|x - v\|^2$$

Example 4 (Projection)

If f is the indicator function of a closed nonempty convex set C, then the x-update is

$$x^+ = \operatorname*{argmin}_x \left(f(x) + \frac{\rho}{2}\|x - v\|_2^2 \right) = \Pi_C(v),$$

where

$$f(x) = \begin{cases} 0 & \text{if } x \in C \\ \infty & \text{otherwise} \end{cases}$$

and Π_C denotes projection onto C.

Example 5 (Soft Thresholding)

For an example that will come up often in the sequel, consider $f(x) = \lambda\|x\|_1$ (with $\lambda > 0$) and $A = I$. In this case, the (scalar) x-update is

$$\begin{aligned} x^+ &= \underset{x}{\text{argmin}} \;\; \lambda|x| + \frac{1}{2}(x - v)^2 \\ &= \begin{cases} v - \lambda & \text{, if } u > \lambda \\ 0 & \text{, if } -\lambda \leq v \leq \lambda \\ v + \lambda & \text{, if } u < -\lambda. \end{cases} \end{aligned}$$

Definition 6 (Soft Thresholding operator)

$S_\lambda : \mathbb{R} \mapsto \mathbb{R}$ is defined by

$$S_\lambda(v) = \begin{cases} v - \lambda & \text{, if } u > \lambda \\ 0 & \text{, if } -\lambda \leq v \leq \lambda \\ v + \lambda & \text{, if } u < -\lambda. \end{cases}$$

Example 7 (Lasso)

The lasso regression estimator is obtained by solving the problem

$$\min_{x \in \mathbb{R}^p} \quad \|Ax - b\|^2 + \lambda \|x\|_1$$

An equivalent problem is given by

$$\begin{aligned} \min_{x,z} \quad & \|Ax - b\|^2 + \lambda \|z\|_1 \\ \text{subject to} \quad & Ix - Iz = 0. \end{aligned}$$

The scaled form of ADMM defines

$$L_\rho(x, z, u) = \|Ax - b\|^2 + \lambda\|z\|_1 + \frac{\rho}{2}\|x - z + u\|^2.$$

The following is the first iteration of the ADMM.

1. Initialize $u^{(0)}$ and $z^{(0)}$.
2. Obtain $x^{(1)} = \underset{x}{\text{argmin}} \; \|Ax - b\|^2 + \frac{\rho}{2}\|x - z^{(0)} + u^{(0)}\|^2$
3. Obtain $z^{(1)} = \underset{z}{\text{argmin}} \; \lambda\|z\|_1 + \frac{\rho}{2}\|x^{(1)} - z + u^{(0)}\|^2$
4. $u^{(1)} = u^{(0)} + x^{(1)} - z^{(1)}$

Computation of the steps 2 and 3

$x^{(1)}$ has a closed form as

$$x^{(1)} = 2(A^\top A + \rho I)^{-1}(2b^\top A\rho(z^{(0)} - u^{(0)}))$$

$z^{(1)}$ has also a closed form. Let $v = (v_1, \cdots, v_n) = x^{(1)} + u^{(0)}$. Note that $\lambda\|z\|_1 + \frac{\rho}{2}\|z - v\|^2 = \lambda\sum_{j=1}^{p}|z_j| + \frac{\rho}{2}\sum_{j=1}^{p}(z_j - v_j)^2$ such that $z^{(1)} = (z_1^{(1)}, \cdots, z_p^{(1)})$ where

$$\begin{aligned} z_j^{(1)} &= \operatorname*{argmin}_z \; \lambda|z| + \frac{\rho}{2}(z - v_j)^2 \\ &= \operatorname*{argmin}_z \; \frac{\lambda}{\rho}|z| + \frac{1}{2}(z - v_j)^2 \\ &= S_{\lambda/\rho}(v_j) \text{ (Soft thresholding operator)} \end{aligned}$$

Consensus Problem

review

What is a consensus problem?

Consider the case with a single global variable x, with the objective and constraint split into N parts:

$$\min_{x} \qquad f(x) = \sum_{i=1}^{N} f_i(x), \tag{5}$$

where $x \in \mathbb{R}^n$, and $f_i : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ are convex and encode constraints by assuming $f_i(X) = +\infty$ when a constraint is violated.

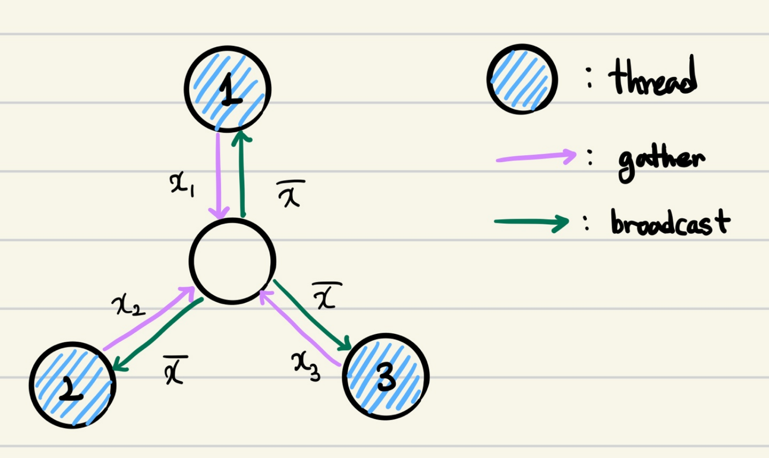

Figure 1: Parallel process in x-udpate

If one processor has high computing complexity, it leads to a bottleneck state.

What is a sharing problem?

The *sharing problem* involves each agent adjusting its variable to minimize its individual cost f_i, as well as the shared objective g.

$$\min_{x_i} \quad \sum_{i=1}^{N} f_i(x_i) + g(\sum_{i=1}^{N} x_i), \tag{6}$$

where $x_i \in \mathbb{R}^n$, f_i is a local cost function in subsystem i and g is the shared objective.

The sharing problem is important because many useful problems can be put into this form.

When can we use distributed fitting?

- This is useful either when there are so many training examples that it is inconvenient or impossible to process them on a single machine or when the data is naturally collected or stored in a distributed fashion.
  ex) online social network data, web server access logs, wireless sensor networks
- This is useful when the data have modest examples but a large of features.
  ex) NLP, bioinformatics

Suppose we have this problem

$$\text{minimize} \qquad l(Ax - b) + r(x) \tag{7}$$

where $\quad x \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m, l : \mathbb{R}^m \to \mathbb{R}, r : \mathbb{R}^n \to \mathbb{R}$

Definition 8 (Data parallelization)

Suppose that $l(u_1, \cdots, u_m) = l_1(u_1) + \cdots + l_m(u_m)$. The problem is written by

$$l(Ax - b) = \sum_{i=1}^{m} l_i(a_i^\top x - b_i),$$

where $a_i^\top$ is the ith row vector of A. If $r(\cdot)$ is separable, it is a consensus and sharing problem.

(7) is a consensus form

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{N} l_i(a_i^\top x_i - b_i) + r(z) && (8) \\ \text{subject to} \quad & x_i - z = 0. && (9) \end{aligned}$$

Then, the scaled Lagrangian of the above problem (8) is obtained by

$$L_\rho(x_1, \ldots, x_N, z, y)$$
$$\sum_{i=1}^{N} (l_i(a_i^\top x_i - b_i) + r(z) + (\rho/2)\|x_i - z + u_i\|_2^2 + \|u_i\|_2^2)$$

The resulting ADMM algorithm is the following:

$$
\begin{aligned}
x_i^{(k+1)} &:= \arg\min_{x_i}(l_i(A_i x_i - b_i) + (\rho/2)\|x_i - z_i^k + u_i^k\|_2^2) \\
z^{(k+1)} &:= \arg\min_{z}(r(z) + (\rho/2)\sum_{i=1}^{N}\|x_i - z + u_i\|_2^2) \\
&= \arg\min_{z}(r(z) + (N\rho/2)\|z - \bar{x}^{(k+1)} - \bar{u}^k\|_2^2) \\
u_i^{(k+1)} &:= u_i^k + x_i^{k+1} - z^{k+1}
\end{aligned}
$$

Example 9 (Parallel computing of Lasso: Data Parallelism)

$$\begin{aligned} \min \quad & \frac{1}{2}\|Ax-b\|_2^2+\lambda\|z\|_1 \\ \text{subject to} \quad & x-z=0, \lambda \geq 0 \end{aligned}$$

Let $A_i \in \mathbb{R}^{n_i \times p}$ for $i=1,\cdots,k$ and $\sum_{i=1}^N n_i = n$ and $A=[A_1^\top,\cdots,A_N^\top]^\top$. Similarly let $b_i \in \mathbb{R}^{n_i}$ and $b=(b_1^\top,\cdots,b_N^\top)^\top$. Then, an equivalent problem is given by

$$\begin{aligned} \min \quad & \frac{1}{2}\sum_{i=1}^N \|A_i x_i - b_i\|_2^2+\lambda\|z\|_1 \\ \text{subject to} \quad & x_i - z = 0 \text{ for } i=1,\cdots,N. \end{aligned}$$

Let $x = (x_1^\top, \cdots, x_N^\top)^\top$, and $u = (u_1^\top, \cdots, u_N^\top)^\top$ and $B = (I, \cdots, I)^\top \in \mathbb{R}^{Np \times p}$, then we can obtain the Scaled Augment Lagrangian form

$$\begin{aligned} L_\rho(x, z, u) &= \frac{1}{2}\|Ax - b\|^2 + \lambda\|z\|_1 + \frac{\rho}{2}\|Ix - Bz + u\|^2 \\ &= \frac{1}{2}\sum_{i=1}^{N}\|A_i x_i - b_i\|^2 + \lambda\|z\|_1 + \frac{\rho}{2}\sum_{i=1}^{N}\|x_i - z + u_i\|^2 \end{aligned}$$

Note that for fixed z and u_i $L_\rho(x, z, u)$ is separable with respect to $x_1, \cdots, x_k$. Thus,

$$\begin{aligned} x_i^{k+1} &:= \arg\min_{x_i}(\frac{1}{2}\|Ax_i - b_i\|_2^2 + (\rho/2)\|x_i - z^k + u_i^k\|_2^2) \\ &= (A_i^\top A_i + \rho I)^{-1}(A_i b_i + \rho(z^k - u_i^k)) \end{aligned}$$

are obtained in each server.

Denote the jth element of x_i and u_i by x_{ij} and u_{ij}. For fixed x_i and u_i for $i = 1, \cdots, k$, L_ρ is separable since

$$\begin{aligned} & \lambda \|z\|_1 + \frac{\rho}{2} \|x - Bz + u\|^2 \\ = \quad & \lambda \sum_{j=1}^{p} |z_j| + \frac{\rho}{2} \sum_{i=1}^{k} \sum_{j=1}^{p} (x_{ij} + u_{ij} - z_j)^2 \\ = \quad & \sum_{j=1}^{p} \left(\lambda |z_j| + \frac{\rho}{2} \sum_{i=1}^{k} (x_{ij} + u_{ij} - z_j)^2 \right) \end{aligned}$$

Let $v_{ij} = x_{ij} + u_{ij}$. The minimizer of $L_\rho(x, z, u)$ for fixed x and u are obtained

$$z_j^+ \quad = \quad \text{argmin}_z \;\; \lambda |z| + \frac{\rho}{2} \sum_{i=1}^{k} (z_j - v_{ij})^2$$

Temporarily omit the index j in z and v.

$$\begin{aligned} & \text{argmin}_z \quad \lambda|z| + \frac{\rho}{2}\sum_{i=1}^{k}(z - v_i)^2 \\ = & \text{argmin}_z \quad \lambda|z| + \frac{\rho}{2}(kz^2 - 2(\sum_{i=1}^{k} v_i)z) \\ = & \text{argmin}_z \quad \lambda|z| + \frac{k\rho}{2}(z - \bar{v})^2, \end{aligned}$$

where $\bar{v} = \sum_{i=1}^{k} v_i/k$. Thus, the minimizer z is obtained by the soft thresholding operator $S_{\lambda/(k\rho)}(\bar{v})$.

The resulting ADMM algorithm is the following:

$$\begin{aligned}
x_i^{(k+1)} &:= \arg\min_{x_i} \frac{1}{2}\|Ax_i - b_i\|_2^2 + (\rho/2)\|x_i - z^k + u_i^k\|^2 \\
&= (A_i^\top A_i + \rho I)^{-1}(A_i b_i + \rho(z^k - u_i^k)) \\
z^{k+1} &:= (z_1^{(k+1)}, \cdots, z_p^{(k+1)}), \\
&\quad \text{where } z_j^{(k+1)} = S_{\lambda/(k\rho)}(\bar{x}_j^{(k+1)} + \bar{u}_j^{(k)}) \\
u_i^{k+1} &:= u_i^k + x_i^{k+1} - z^{k+1}
\end{aligned}$$

Note that $(A_i^\top A_i + \rho I)^{-1}$ and b_i for each i do not depend on the updated solutions. Thus, it would be beneficial to restore these quantities in each memory.

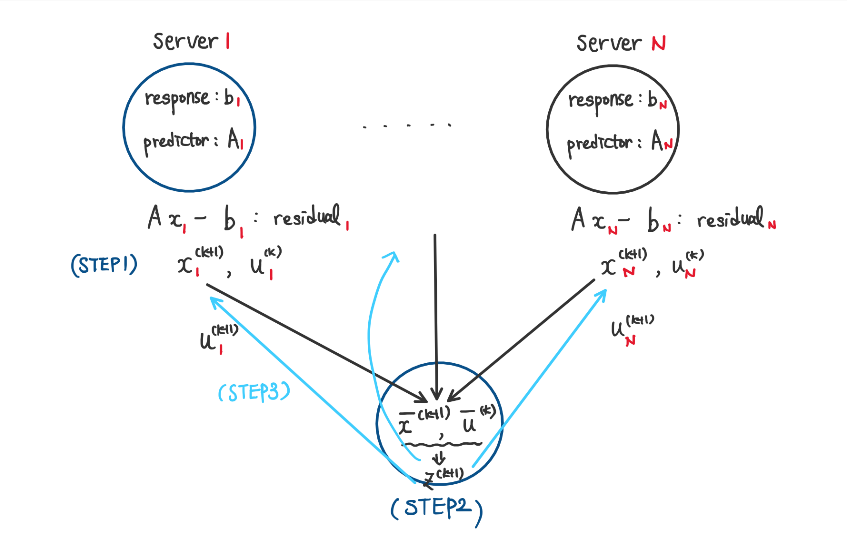

Figure 2: Flow of Computation

matrix Inversion lemma

when we proceed in updating x_i, we have to find $(A_i^\top A_i + \rho I)^{-1}$ value then, we can use matrix Inversion lemma,

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}$$

if A and C is identity matrix, then,

$$(I + UV)^{-1} = I - U(I + VU)^{-1}V$$

Definition 10 (Feature Parallelization)

Suppose we have this problem

$$\min \quad l(\sum_{i=1}^{N} A_i x_i - b) + \sum_{i=1}^{N} r_i(x_i)$$

Following the approach used for the sharing problem, we express the problem as

$$\begin{aligned} \min \quad & l(\sum_{i=1}^{N} z_i - b) + \sum_{i=1}^{N} r_i(x_i) \\ \text{subject to} \quad & A_i x_i - z_i = 0, i = 1, 2 \ldots N. \end{aligned}$$

The resulting ADMM algorithm is the following:

$$\begin{aligned}
x_i^{k+1} &:= \arg\min_{x_i}(r_i(x_i)) + (\rho/2)\|A_i x_i - z_i^k + u_i^k\|_2^2) \\
z_i^{k+1} &:= \arg\min_{z}(l(\sum_{i=1}^{N} z_i - b) + \sum_{i=1}^{N}(\rho/2)\|A_i x_i^{k+1} - z_i^k + u_i^k\|_2^2) \\
u_i^{k+1} &:= u_i^k + A_i x_i^{k+1} - z_i^{k+1}
\end{aligned}$$

Updating z_i is simplified by two steps. Note that

$$\begin{aligned}\sum_{i=1}^{N}\|z_i - c_i\|^2 &= \sum_{i=1}^{N}\|z_i - c_i - \bar{z} + \bar{c} + \bar{z} - \bar{c}\|^2 \\ &= \sum_{i=1}^{N}\|z_i - c_i - \bar{z} + \bar{c}\|^2 + \sum_{i=1}^{N}\|\bar{z} - \bar{c}\|^2 \\ &\geq N\|\bar{z} - \bar{c}\|^2,\end{aligned}$$

where $\bar{z} = \sum_{i=1}^{N} z_i$ and $\bar{c} = \sum_{i=1}^{N} c_i/N$. The second equality holds for $\sum_{i=1}^{N}(z_i - c_i - \bar{z} + \bar{c})^\top(\bar{z} - \bar{c}) = 0$.

Thus,

$$\begin{aligned} & l(\sum_{i=1}^{N} z_i - b_i) + (\rho/2)\sum_{i=1}^{N} \|z_i - A_i x_i - u_i\|^2 \\ \geq \quad & l(N\bar{z} - b_i) + (N\rho/2)\|\bar{z} - \overline{Ax} - \bar{u}\|, \end{aligned} \tag{10}$$

where $\overline{Ax} = \frac{1}{N}\sum_{i=1}^{N} A_i x_i$, $\overline{u} = \frac{1}{N}\sum_{i=1} \overline{u_i}$.

$$\begin{aligned} \overline{z}^{(k+1)} \quad &:= \quad \arg\min_{\bar{z}}(l(N\bar{z} - b) + \frac{N\rho}{2}\|\bar{z} - \overline{Ax}^{(k+1)} - \bar{u}^k\|_2^2) \\ z_i^{(k+1)} \quad &:= \quad \overline{z}^{(k+1)} + A_i x_i^{k+1} + u_i^k - \overline{Ax}^{(k+1)} - \bar{u}^k. \end{aligned}$$

We investigate the updating rule

$$u_i^{(k+1)} = u_i^{(k)} + A_i x_i^{(k+1)} - z_i^{(k+1)}$$

(See the unconstrained optimization slide 1 for Jacobian of composite function)

Let $z = (z_1^\top, \cdots, z_N^\top)^\top \in \mathbb{R}^{nN}$ and I_n be $n \times n$ identity matrix and $C = [I_n, \cdots, I_n] \in \mathbb{R}^{n \times (nN)}$. Denote $h : z \in \mathbb{R}^{nN} \mapsto Cz \in \mathbb{R}^m$ then we can write

$$l(\sum_{i=1}^{N} z_i) = l(Cz) = (l \circ h)(z)$$

Since the Jacobian of h and $l \circ h$ are A and $J_l(h(x))J_h(x)$,

$$\frac{\partial l(h(z))}{\partial z} = C^\top \nabla l(Cz)$$

$$C^\top \nabla l(Cz) = \begin{pmatrix} I_n \\ \vdots \\ I_n \end{pmatrix} \nabla l(Cz) = \begin{pmatrix} \nabla l(Cz) \\ \vdots \\ \nabla l(Cz) \end{pmatrix}$$

Thus, updating $z^{(k+1)}$ in the ADMM implies that

$$\frac{\partial L_\rho(z^{(k+1)})}{\partial z_i} = \nabla l(Cz^{(k+1)}) + \rho(z_i^{(k+1)} - A_i x_i^{(k+1)} + u_i^{(k)}) = 0$$

for $i = 1, \cdots, N$.

In addition, the dual feasibility of the Lagrangian is defined by

$$\frac{\partial L(z^{(k+1)})}{\partial z_i} = \nabla l(Cz^{(k+1)}) + \nu_i^{(k+1)} = l(Cz^{(k+1)}) + \rho u_i^{(k+1)} = 0$$

Thus,

$$\rho u_i^{(k+1)} = \rho(z_i^{(k+1)} - A_i x_i^{(k+1)} + u_i^{(k)})$$

implies the dual feasibility of the Lagrangian. That is, the updating

$$u_i^{(k+1)} = u_i^{(k)} + z_i^{(k+1)} - A_i x_i^{(k+1)}$$

is given by the admm. Note that equally

$$u_i^{(k+1)} = \bar{A}x^{(k+1)} + \bar{u}^{(k)} - \bar{z}^{(k+1)}.$$

Hence, we can get The resulting ADMM algorithm

$$\begin{aligned} x_i^{k+1} &:= \arg\min_{x_i}(r_i(x_i)) + (\rho/2)\|A_i x_i - A_i x_i^k - \bar{z}^k + \bar{A}x^k + u^k\|_2^2) \\ \bar{z}^{k+1} &:= \arg\min_{z}(l(N\bar{z} - b) + \sum_{i=1}^{N}(\rho/2)\|\bar{z} - \bar{A}x^{k+1} - u^k\|_2^2) \\ u^{k+1} &:= u^k + \bar{A}x^{k+1} - \bar{z}^{k+1} \end{aligned}$$

Example 11 (Lasso)

If we separate the lasso problem based on variables,

$$\min \frac{1}{2}\|\sum_{i=1}^{N} A_i x_i - b\|_2^2 + \lambda \sum_{i=1}^{N} \|x_i\|_1$$

Then, we can change $A_i x_i = z_i$ for applying to ADMM,

$$\begin{array}{rl} \min & \frac{1}{2}\|\sum_{i=1}^{N} z_i - b\|_2^2 + \lambda \sum_{i=1}^{N} \|x_i\|_1 \\ \text{subject to} & A_i x_i - z_i = 0 \text{ for all } i \end{array}$$

We can obtain the Scaled Augment Lagrangian form,

$$L(x_i, z_i, u_i) = \frac{1}{2}\|\sum_{i=1}^{N} z_i - b\|_2^2 + \lambda \sum_{i=1}^{N} \|x_i\|_1 + \frac{\rho}{2}\|A_i x_i - z_i + u_i\|_2^2 + \frac{\rho}{2}\|u\|_2^2$$

The resulting ADMM algorithm is the following:

$$\begin{aligned}
x_i^{k+1} &:= \arg\min_{x_i}(\frac{\rho}{2}\|A_i x_i - A_i x_i^k - \bar{z}^k + \bar{A}x^k + u^k\|_2^2 + \lambda\|x_i\|_1) \\
\bar{z}^{k+1} &:= \frac{1}{N+\rho}(b + \rho\bar{A}x^{k+1} + \rho u^k) \\
u^{k+1} &:= u^k + \bar{A}x^{k+1} - \bar{z}^{k+1}
\end{aligned}$$

In the x_i updates, we have $x_i^{k+1} = 0$ if and only if,

$$\|A_i^\top (A_i x_i^k + \bar{z}^k - \bar{Ax}^k - u^k)\|_2 \leq \lambda/\rho$$

when this occurs, the x_i updates is fast.

Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., et al. (2011).
Distributed optimization and statistical learning via the alternating direction method of multipliers.
Foundations and Trends® in Machine learning, 3(1):1–122.