

Multi-block ADMM Methods for Linear Programming

Nico Chaves, Junjie (Jason) Zhu

December 7, 2016

1 Introduction

The alternating direction method of multipliers (ADMM) is a first-order method that can be used to solve various optimization problems. It splits the original problem into smaller pieces that are easier to handle during each iterative update. ADMM relies on the idea of the augmented Lagrangian method to robustify the problem and the splitting procedure makes it a proximal method that can be adopted in a wide range of optimization problems.

The goal of this project is to use ADMM to solve the primal linear programming (LP) problem:

$$\begin{aligned} & \text{minimize}_{\mathbf{x}} && \mathbf{c}^T \mathbf{x} \\ & \text{subject to} && A\mathbf{x} = \mathbf{b}, \\ & && \mathbf{x} \geq \mathbf{0} \end{aligned} \tag{OPT1}$$

or its dual

$$\begin{aligned} & \text{maximize}_{\mathbf{y}, \mathbf{s}} && \mathbf{b}^T \mathbf{y} \\ & \text{subject to} && A^T \mathbf{y} + \mathbf{s} = \mathbf{c}, \\ & && \mathbf{s} \geq \mathbf{0}. \end{aligned} \tag{OPT2}$$

In this report, we first re-formulate the problems to solve for closed form updates on the primal and the dual LP problems. We alternatively consider the same respective formulation using the interior-point method. Along with these different kinds of implementations, we are specifically interested in the effect of preconditioning and block-splitting on the convergence rate and the accuracy of the solvers. We implemented all the algorithms in MATLAB and ran extensive experiments to test how these methods perform on different simulated problems.

Preconditioning

For the analysis on our various implementations of ADMM, we are also interested in the impact of preconditioning the problem. Pre-conditioning involves computing

$$A' = (AA^T)^{-1/2}A, \quad \mathbf{b}' = (AA^T)^{-1/2}A\mathbf{b}', \tag{1}$$

and substituting A' and \mathbf{b}' for A and \mathbf{b} in the original problems (OPT1) and (OPT2). Clearly, this re-formulation does not change the feasible regions for (OPT1) or (OPT2).

Block-Splitting ADMM

ADD MORE INTRO ABOUT BLOCK SPLITTING

Outline of Report

ADD MORE INFO - JASON

2 Basic ADMM for LP

Primal Re-formulation

We re-formulate the primal LP problem as:

$$\begin{aligned} & \text{minimize}_{\mathbf{x}_1, \mathbf{x}_2} && \mathbf{c}^T \mathbf{x}_1 \\ & \text{subject to} && A\mathbf{x}_1 = \mathbf{b} \\ & && \mathbf{x}_1 - \mathbf{x}_2 = \mathbf{0} \\ & && \mathbf{x}_2 \geq \mathbf{0} \end{aligned} \tag{OPT3}$$

and consider the split Lagrangian function:

$$L^P(\mathbf{x}_1, \mathbf{x}_2, \mathbf{y}, \mathbf{s}) = \mathbf{c}^T \mathbf{x}_1 - \mathbf{y}^T (A\mathbf{x}_1 - \mathbf{b}) - \mathbf{s}^T (\mathbf{x}_1 - \mathbf{x}_2) + \frac{\beta}{2} \left(\|A\mathbf{x}_1 - \mathbf{b}\|^2 + \|\mathbf{x}_1 - \mathbf{x}_2\|^2 \right).$$

Primal Update

The updates for the primal variables in ADMM requires computing the gradient of these variables and setting them to zero. Here we update \mathbf{x}_1 and \mathbf{x}_2 sequentially using the update function for each respectively:

$$\mathbf{x}_1^{k+1} = \arg \min_{\mathbf{x}_1^k} L^P(\mathbf{x}_1^k, \mathbf{x}_2^k, \mathbf{y}^k, \mathbf{s}^k), \tag{2}$$

$$\mathbf{x}_2^{k+1} = \arg \min_{\mathbf{x}_2^k \geq 0} L^P(\mathbf{x}_1^{k+1}, \mathbf{x}_2^k, \mathbf{y}^k, \mathbf{s}^k). \tag{3}$$

To solve for \mathbf{x}_1 , we set the gradient to zero:

$$\nabla_{\mathbf{x}_1} L^P(\mathbf{x}_1, \mathbf{x}_2, \mathbf{y}, \mathbf{s}) = \mathbf{c} - A^T \mathbf{y} - \mathbf{s} + \beta (A^T (A\mathbf{x}_1 - \mathbf{b}) + (\mathbf{x}_1 - \mathbf{x}_2)) = \mathbf{0}$$

and we obtain the update step for \mathbf{x}_1 :

$$\mathbf{x}_1^{k+1} = (A^T A + I)^{-1} \left(\frac{1}{\beta} A^T \mathbf{y}^k + \frac{1}{\beta} \mathbf{s}^k - \frac{1}{\beta} \mathbf{c} + A^T \mathbf{b}^k + \mathbf{x}_2^k \right) \tag{4}$$

Similarly, we solve the gradient

$$\nabla_{\mathbf{x}_2} L^P = \mathbf{s} + \beta (\mathbf{x}_2 - \mathbf{x}_1) = \mathbf{0}$$

to obtain the update for \mathbf{x}_2 . Since the elements of \mathbf{x}_2 are separable, we can update each one independently as follows:

$$x_{2,j}^{k+1} = \max \left\{ x_{1,j}^k - \frac{1}{\beta} s_j^k, 0 \right\} \text{ for } j = 1, \dots, n. \tag{5}$$

Next, we update the multipliers \mathbf{y} and \mathbf{s} using

$$\mathbf{y}^{k+1} = \mathbf{y}^k + \beta (A\mathbf{x}_1^{k+1} - \mathbf{b}) \tag{6}$$

$$\mathbf{s}^{k+1} = \mathbf{s}^k - \beta (\mathbf{x}_1^{k+1} - \mathbf{x}_2^{k+1}) \tag{7}$$

These two updates guarantee that $(\mathbf{x}_1^{k+1}, \mathbf{x}_2^{k+1}, \mathbf{y}^{k+1}, \mathbf{s}^{k+1})$ dual feasible, i.e.,

$$\nabla_{\mathbf{x}_1, \mathbf{x}_2} L^P(\mathbf{x}_1^k, \mathbf{x}_2^k, \mathbf{y}^k, \mathbf{s}^k) = \mathbf{0}.$$

We terminate when the solution primal feasible, i.e., $A\mathbf{x}_1^{k+1} - \mathbf{b} \rightarrow \mathbf{0}$ and $\mathbf{x}_1^{k+1} - \mathbf{x}_2^{k+1} \rightarrow \mathbf{0}$.

Dual Re-formulation

We convert the problem of the dual to a minimization problem to apply ADMM:

$$\begin{aligned} & \underset{\mathbf{y}, \mathbf{s}}{\text{minimize}} && -\mathbf{b}^T \mathbf{y} \\ & \text{subject to} && A^T \mathbf{y} + \mathbf{s} = \mathbf{c}, \\ & && \mathbf{s} \geq \mathbf{0}. \end{aligned} \tag{OPT4}$$

The dual LP problem does not require additional re-formulation because the non-zero variable \mathbf{s} is already separable, which can be observed via the dual Lagrangian function:

$$L^d(\mathbf{y}, \mathbf{s}, \mathbf{x}) = -\mathbf{b}^T \mathbf{y} - \mathbf{x}^T (A^T \mathbf{y} + \mathbf{s} - \mathbf{c}) + \frac{\beta}{2} \|A^T \mathbf{y} + \mathbf{s} - \mathbf{c}\|^2.$$

Note that \mathbf{x} would be non-positive since we changed maximization to minimization and we would need to take the negative of \mathbf{x} of the final solution as the solution to the original LP.

Dual Update

Similar to the primal updates, we use the updates for the primal variables \mathbf{y} and \mathbf{s} according to

$$\mathbf{y}^{k+1} = \arg \min_{\mathbf{y}} L^d(\mathbf{y}^k, \mathbf{s}^k, \mathbf{x}^k), \tag{8}$$

$$\mathbf{s}^{k+1} = \arg \min_{\mathbf{s} \geq \mathbf{0}} L^d(\mathbf{y}^{k+1}, \mathbf{s}^k, \mathbf{x}^k). \tag{9}$$

Setting the gradients with respect to \mathbf{y} and \mathbf{s} to zero, i.e.,

$$\nabla_{\mathbf{y}} L^d(\mathbf{y}, \mathbf{s}, \mathbf{x}) = -\mathbf{b} - A\mathbf{x} + \beta A (A^T \mathbf{y} + \mathbf{s} - \mathbf{c}) = \mathbf{0}, \tag{10}$$

$$\nabla_{\mathbf{s}} L^d(\mathbf{y}, \mathbf{s}, \mathbf{x}) = -\mathbf{x} + \beta (A^T \mathbf{y} + \mathbf{s} - \mathbf{c}) = \mathbf{0}, \tag{11}$$

we have the updates for \mathbf{y} (least-squares problem) and \mathbf{s} :

$$\mathbf{y}^{k+1} = (AA^T)^{-1} \left(\frac{1}{\beta} (A\mathbf{x}^k + \mathbf{b}) - A\mathbf{s}^k + A\mathbf{c} \right), \tag{12}$$

$$s_j^{k+1} = \max \left\{ \frac{1}{\beta} x_j^k - (A^T \mathbf{y}^{k+1})_j + c_j, 0 \right\}. \tag{13}$$

We can update the multipliers \mathbf{x} by

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \beta (A^T \mathbf{y}^{k+1} + \mathbf{s}^{k+1} - \mathbf{c}). \tag{14}$$

The termination condition for this dual would be $A\mathbf{x}^{k+1} + \mathbf{b} \rightarrow \mathbf{0}$.

3 Interior-Point ADMM for LP

Primal LP Updates

We can use the previous formulation in (OPT3) with \mathbf{x}_1 and \mathbf{x}_2 using the barrier function:

$$\begin{aligned} & \underset{\mathbf{x}_1, \mathbf{x}_2}{\text{minimize}} && \mathbf{c}^T \mathbf{x}_1 + \mu \sum_j \ln((x_{2,j}) \\ & \text{subject to} && A\mathbf{x}_1 = \mathbf{b}, \\ & && \mathbf{x}_1 - \mathbf{x}_2 = \mathbf{0}, \\ & && \mathbf{x}_2 > \mathbf{0}, \end{aligned} \tag{OPT5}$$

where the Lagrangian function is given by

$$L_\mu^p(\mathbf{x}_1, \mathbf{x}_2, \mathbf{y}, \mathbf{s}) = \mathbf{c}^T \mathbf{x}_1 - \mu \sum_j \ln(x_{2,j}) - \mathbf{y}^T (A\mathbf{x}_1 - \mathbf{b}) - \mathbf{s}^T (\mathbf{x}_1 - \mathbf{x}_2) + \frac{\beta}{2} (\|A\mathbf{x}_1 - \mathbf{b}\|^2 + \|\mathbf{x}_1 - \mathbf{x}_2\|^2).$$

Setting the gradient of this to zero yields the same update for \mathbf{x}_1 as in (4). However, the we need to modify the update for \mathbf{x}_2 as now we have:

$$s_j - \frac{\mu}{x_{2,j}} + \beta(x_{2,j} - x_{1,j}) = 0 \quad (15)$$

which yields

$$x_{2,j} = \frac{1}{2\beta} \left(\beta x_{1,j} - s_j + \sqrt{\beta^2 x_{1,j}^2 - 2\beta s_j x_{1,j} + s_j^2 + 4\beta\mu} \right), \text{ for } j = 1, \dots, n \quad (16)$$

The update equation for \mathbf{y} and \mathbf{s} remain the same as in (6) and (7). The update equation for \mathbf{x} remain the same as in (14). At the end of each iteration, we update $\mu^{k+1} = \gamma\mu^k$ for some constant γ , and the terminal condition remains the same as the non-interior-point method.

Dual LP Updates

We can use the previous formulation in (OPT4) using the barrier function:

$$\begin{aligned} \text{minimize}_{\mathbf{y}, \mathbf{s}} \quad & -\mathbf{b}^T \mathbf{y} + \mu \sum_j \ln(s_j) \\ \text{subject to} \quad & A^T \mathbf{y} + \mathbf{s} = \mathbf{c}, \\ & \mathbf{s} > \mathbf{0}. \end{aligned} \quad (\text{OPT6})$$

where the Lagrangian function is given by

$$L_\mu^d(\mathbf{y}, \mathbf{s}, \mathbf{x}) = -\mathbf{b}^T \mathbf{y} - \mu \sum_j \ln(s_j) - \mathbf{x}^T (A^T \mathbf{y} + \mathbf{s} - \mathbf{c}) + \frac{\beta}{2} \|A^T \mathbf{y} + \mathbf{s} - \mathbf{c}\|^2.$$

Setting the gradient of this to zero yields the same update for \mathbf{y} as in (12) However, the we need to modify the update for \mathbf{s} as now we have:

$$-\frac{\mu}{s_j} + x_j + \beta(s_j - (\mathbf{c} - (A^T \mathbf{y}))_j) = 0 \quad (17)$$

which yields

$$s_j = \frac{1}{2\beta} \left(\beta(\mathbf{c} - (A^T \mathbf{y}))_j + x_j + \sqrt{\beta^2(\mathbf{c} - (A^T \mathbf{y}))_j^2 - 2\beta(\mathbf{c} - (A^T \mathbf{y}))_j x_j + x_j^2 + 4\beta\mu} \right), \text{ for } j = 1, \dots, n \quad (18)$$

The update equation for \mathbf{x} remain the same as in (14). At the end of each iteration, we update $\mu^{k+1} = \gamma\mu^k$ for some constant γ , and the terminal condition remains the same as the non-interior-point method.

Multi-Block ADMM

Primal with Block Splitting

For simplicity, we first consider splitting the problem into 2 blocks of equal size as follows:

$$\mathbf{x}_1 = \begin{bmatrix} \mathbf{x}_{1,1} \\ \mathbf{x}_{1,2} \end{bmatrix}$$

$$A = \begin{bmatrix} A_1 & A_2 \end{bmatrix}$$

Then, the Lagrangian can be expressed as:

$$\begin{aligned} L^P(\mathbf{x}_1, \mathbf{x}_2, \mathbf{y}) &= \mathbf{c}_1^T \mathbf{x}_{1,1} + \mathbf{c}_2^T \mathbf{x}_{1,2} - \mathbf{y}^T (A_1 \mathbf{x}_{1,1} + A_2 \mathbf{x}_{1,2} - \mathbf{b}) - \mathbf{s}_1^T (\mathbf{x}_{1,1} - \mathbf{x}_{2,1}) - \mathbf{s}_2^T (\mathbf{x}_{1,2} - \mathbf{x}_{2,2}) \\ &\quad + \frac{\beta}{2} \left(\|A_1 \mathbf{x}_{1,1} + A_2 \mathbf{x}_{1,2} - \mathbf{b}\|^2 + \|\mathbf{x}_{1,1} - \mathbf{x}_{2,1}\|^2 + \|\mathbf{x}_{1,2} - \mathbf{x}_{2,2}\|^2 \right) \end{aligned}$$

Taking the gradient with respect to the 1st block of \mathbf{x}_1 :

$$\nabla_{\mathbf{x}_{1,1}} L^P(\mathbf{x}_1, \mathbf{x}_2, \mathbf{y}) = \mathbf{c}_1 - A_1^T \mathbf{y} - \mathbf{s}_1 + \beta (A_1^T (A_1 \mathbf{x}_{1,1} + A_2 \mathbf{x}_{1,2} - \mathbf{b}) + (\mathbf{x}_{1,1} - \mathbf{x}_{2,1}))$$

Setting the gradient to 0, we obtain the update for $\mathbf{x}_{1,1}$:

$$\mathbf{x}_{1,1}^{k+1} = (A_1^T A_1 + I)^{-1} \left(\frac{1}{\beta} A_1^T \mathbf{y} + \frac{1}{\beta} \mathbf{s}_1 - \frac{1}{\beta} \mathbf{c}_1 + A_1^T \mathbf{b} + \mathbf{x}_{2,1}^k - A_1^T A_2 \mathbf{x}_{1,2}^k \right)$$

By symmetry, the update step for the 2nd block of \mathbf{x}_1 is:

$$\mathbf{x}_{1,2}^{k+1} = (A_2^T A_2 + I)^{-1} \left(\frac{1}{\beta} A_2^T \mathbf{y} + \frac{1}{\beta} \mathbf{s}_2 - \frac{1}{\beta} \mathbf{c}_2 + A_2^T \mathbf{b} + \mathbf{x}_{2,2}^k - A_2^T A_1 \mathbf{x}_{1,1}^{k+1} \right)$$

Note that here we use $\mathbf{x}_{1,1}^{k+1}$ to update $\mathbf{x}_{1,2}$. If we use a randomized update order, then we may end up updating $\mathbf{x}_{1,2}$ first instead.

Block Splitting (for a general number of blocks)

We can easily extend the above result to a general number of blocks. Let U denote the set of blocks which have already been updated at the current iteration. To update block i of \mathbf{x}_1 , we compute:

$$\mathbf{x}_{1,i}^{k+1} = (A_i^T A_i + I)^{-1} \left(\frac{1}{\beta} A_i^T \mathbf{y} + \frac{1}{\beta} \mathbf{s}_i - \frac{1}{\beta} \mathbf{c}_i + A_i^T \mathbf{b} + \mathbf{x}_{2,i}^k - \sum_{j \neq i, j \in U} A_i^T A_j \mathbf{x}_{1,j}^{k+1} - \sum_{j \neq i, j \notin U} A_i^T A_j \mathbf{x}_{1,j}^k \right)$$

where A_i refers to the i^{th} block of columns of A .

Dual ADMM

$$L^d(\mathbf{y}, \mathbf{s}, \mathbf{x}) = -\mathbf{b}^T \mathbf{y} - \mathbf{x}^T (A^T \mathbf{y} + \mathbf{s} - \mathbf{c}) + \frac{\beta}{2} \|A^T \mathbf{y} + \mathbf{s} - \mathbf{c}\|^2$$

Dual ADMM With Block Splitting

For simplicity, we first consider splitting the problem into 2 blocks of equal size as follows:

$$\mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}$$

$$\mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}$$

$$A^T = \begin{bmatrix} A_1^T & A_2^T \end{bmatrix}$$

Then the Lagrangian can be expressed as:

$$L^d(\mathbf{y}, \mathbf{s}, \mathbf{x}) = -\mathbf{b}_1^T \mathbf{y}_1 - \mathbf{b}_2^T \mathbf{y}_2 - \mathbf{x}^T (A_1^T \mathbf{y}_1 + A_2^T \mathbf{y}_2 + \mathbf{s} - \mathbf{c}) + \frac{\beta}{2} \|A_1^T \mathbf{y}_1 + A_2^T \mathbf{y}_2 + \mathbf{s} - \mathbf{c}\|^2$$

Differentiating with respect to \mathbf{y}_1 :

$$\nabla_{\mathbf{y}_1} L^d(\mathbf{y}, \mathbf{s}, \mathbf{x}) = -\mathbf{b}_1 - A_1 \mathbf{x} + \beta A_1 (A_1^T \mathbf{y}_1 + A_2^T \mathbf{y}_2 + \mathbf{s} - \mathbf{c})$$

Setting the gradient to 0, we obtain the update for \mathbf{y}_1 :

$$\begin{aligned} A_1 (A_1^T \mathbf{y}_1^{k+1} + A_2^T \mathbf{y}_2^k + \mathbf{s} - \mathbf{c}) &= \frac{1}{\beta} (A_1 \mathbf{x} + \mathbf{b}_1) \\ A_1 A_1^T \mathbf{y}_1^{k+1} &= \frac{1}{\beta} (A_1 \mathbf{x} + \mathbf{b}_1) - A_1 (A_2^T \mathbf{y}_2^k + \mathbf{s} - \mathbf{c}) \\ \mathbf{y}_1^{k+1} &= (A_1 A_1^T)^{-1} \left(\frac{1}{\beta} (A_1 \mathbf{x} + \mathbf{b}_1) - A_1 (A_2^T \mathbf{y}_2^k + \mathbf{s} - \mathbf{c}) \right) \end{aligned}$$

By symmetry, the update for \mathbf{y}_2 is:

$$\mathbf{y}_2^{k+1} = (A_2 A_2^T)^{-1} \left(\frac{1}{\beta} (A_2 \mathbf{x} + \mathbf{b}_2) - A_2 (A_1^T \mathbf{y}_1^{k+1} + \mathbf{s} - \mathbf{c}) \right)$$

assuming that we update \mathbf{y}_2 after \mathbf{y}_1 .

Dual Block Splitting (for a general number of blocks)

We can easily extend the above result to a general number of blocks. Let U denote the set of blocks which have already been updated at the current iteration. To update block i of \mathbf{y} :

$$\mathbf{y}_i^{k+1} = (A_i A_i^T)^{-1} \left(\frac{1}{\beta} (A_i \mathbf{x} + \mathbf{b}_i) - \sum_{j \neq i, j \in U} A_i A_j^T \mathbf{y}_j^{k+1} - \sum_{j \neq i, j \notin U} A_i A_j^T \mathbf{y}_j^k - A_i (\mathbf{s} - \mathbf{c}) \right)$$

where A_i refers to rows of A .