A Distributed Strategy for Computing Proximity Operators

F. Abboud[1,2], E. Chouzenoux[1], J.-C. Pesquet[1], J.-H. Chenot[2], and L. Laborelli[2]
[1] Université Paris-Est, LIGM, UMR CNRS 8049, 77454 Champs sur Marne, France
[2] INA, Institut National de l'Audiovisuel. 94366 Bry sur Marne, France

***Abstract*—Various recent iterative optimization methods require to compute the proximity operator of a sum of functions. We address this problem by proposing a new distributed algorithm for a sum of non-necessarily smooth convex functions composed with arbitrary linear operators. In our approach, each function is associated with a node of a graph, which communicates with its neighbors. Our algorithm relies on a primal-dual splitting strategy that avoids to invert any linear operator, thus making it suitable for processing high-dimensional datasets. The proposed algorithm has a wide array of applications in signal/image processing and machine learning and its convergence is established.**

I. Introduction

Proximal splitting approaches for finding the minimizer of a sum of convex non-necessarily smooth functions have encountered growing popularity and attracted a large interest in the last years [1], [2]. In these approaches, the functions are processed either via their gradient or their proximity operator depending on their differentiability properties. However, when these functions are very complex, a closed form expression of the latter proximity operator does not exist, so that one has to resort to iterative strategies in order to compute it.
Primal-dual splitting methods are of prominent use when dealing with convex optimization problems where numerous linear operators are involved [3], [4], [5], [6]. Indeed, solving both primal and dual problems leads to algorithms that avoid the inversion of any involved linear operator, so making this class of methods very well suited to large-scale problems encountered in various application fields [7], [8]. Primal-dual techniques are based on several well-known strategies such as the Forward-Backward iteration [9], [10], the Douglas-Rachford algorithm [11], [12], or the Alternating Direction Method of Multipliers [13], [14]. Recently, primal-dual algorithms have been combined with block-coordinate approach, where at each iteration only a few blocks are activated following a specific selection rule, and the associated variables are updated [15], [16], [17]. Stochastic and deterministic versions of these algorithms have been used in numerous fields such as image processing and machine learning, where they are often known as dual ascent methods [18], [19]. New interesting algorithms have been produced with a high flexibility in the rule of selection of the blocks and a faster convergence speed, combined with a reduced memory requirement.

All the aforementioned algorithms are designed to be implemented in a centralized manner which may be suboptimal when dealing with massive datasets. Therefore, various asynchronous or distributed extensions have been proposed [13], [20], [14], [21], where each term is handled by a processing unit and an aggregate solution of the optimization problem is reached thanks to communications between those processing units.

In our paper, we propose a new distributed algorithm for computing the proximity operator of the following sum of functions:

$$(\forall x \in \mathbb{R}^N) \qquad H(x) = \sum_{j=1}^{J} h_j(A_j x), \tag{1}$$

where, for every $j \in \{1, \ldots, J\}$, $h_j \colon \mathbb{R}^{M_j} \to \,]-\infty, +\infty]$ are convex possibly nonsmooth functions and A_j is a matrix in $\mathbb{R}^{M_j \times N}$.

The proposed algorithm extends the dual block preconditioned forward-backward algorithm that was recently proposed in [22] to a distributed asynchronous version. Then, each function h_j is now considered as locally related to a node of a connected hypergraph where communications are allowed between neighbors nodes. Moreover, our method takes advantage of variable metric techniques that have been shown to be efficient for accelerating the convergence speed of proximal approaches [23], [24], [25]. Finally, it takes advantage of all the benefits of primal-dual splitting strategies (handling a finite sum of convex functions without inverting none of the involved linear operators) and it offers convergence guaranties.

The reminder of this paper is organized as follows: in Section II we recall some fundamental background and we introduce a centralized dual forward-backward algorithm for computing the desired proximity operator. Section III presents a distributed version of this algorithm, furthermore a parallel variant of the distributed algorithm is proposed. Finally, some conclusions are given.

II. Problem formulation

A. Optimization Tools

Let $\Gamma_0(\mathbb{R}^N)$ designate the class of proper lower-semicontinuous convex functions from $\mathbb{R}^N$ to $]-\infty, +\infty]$ and let $B \in \mathbb{R}^{N \times N}$ be a symmetric positive definite matrix. The proximity operator of $\psi \in \Gamma_0(\mathbb{R}^N)$ at $\widetilde{x} \in \mathbb{R}^N$ relative to the metric induced by B is denoted by $\operatorname{prox}_{B,\psi}(\widetilde{x})$ and defined as the unique solution to the following minimization problem [1]:

$$\underset{x\in\mathbb{R}^N}{\text{minimize}} \;\; \psi(x) + \frac{1}{2}\|x-\widetilde{x}\|_B^2, \tag{2}$$

where the weighted norm $\|\cdot\|_B$ is defined by $\langle \cdot \mid B\, \cdot\rangle^{1/2}$ with $\langle\cdot\mid\cdot\rangle$ the usual scalar product of $\mathbb{R}^N$. When B is set to the identity matrix, note that the standard proximity operator is recovered.

We also define the conjugate of a function $\psi \in \Gamma_0(\mathbb{R}^N)$ as

$$\psi^* : \mathbb{R}^N \to [-\infty, +\infty] : x \to \sup_{\nu\in\mathbb{R}^N} \left(\langle \nu | x\rangle - \psi(\nu)\right). \tag{3}$$

According to the Moreau decomposition theorem [26]:

$$\text{prox}_{B,\psi^*} = \text{Id} - B^{-1}\text{prox}_{B^{-1},\psi}(B\,\cdot). \tag{4}$$

B. Minimization Problem

In this paper, we are interested in computing the proximity operator of H defined in (1) at $\widetilde{x} \in \mathbb{R}^N$, which amounts to find the solution to the following minimization problem:

$$\begin{aligned}\text{Find }\; \widehat{x} &= \text{prox}_H(\widetilde{x}) \\ &= \underset{x\in\mathbb{R}^N}{\text{argmin}} \;\; \sum_{j=1}^{J} h_j(A_j x) + \frac{1}{2}\|x-\widetilde{x}\|^2.\end{aligned} \tag{5}$$

where $\bigcap_{j\in\{1,\ldots,J\}} \text{dom}\,(h_j \circ A_j) \neq \varnothing$. A number of primal-dual algorithms can be applied to solve Problem (5) by resorting to its dual formulation given by:

$$\text{Find }\; \widehat{y} = \underset{y=(y^j)_{1\leqslant j\leqslant J}\in\mathbb{R}^M}{\text{argmin}} \;\; \frac{1}{2}\Big\|\widetilde{x} - \sum_{j=1}^{J} A_j^\top y^j\Big\|^2 + \sum_{j=1}^{J} h_j^*(y^j), \tag{6}$$

where $(h_j^*)_{1\leqslant j\leqslant J}$ are the conjugate functions of $(h_j)_{1\leqslant j\leqslant J}$.

Among existing efficient primal-dual approaches, the Dual Block Preconditioned Forward-Backward algorithm was recently proposed in [22] :

Algorithm 1 Dual Block Preconditioned Forward-Backward

Initialization:
$B_j \in \mathbb{R}^{M_j\times M_j}$ with $B_j \succeq A_j A_j^\top, \quad \forall j \in \{1,\ldots,J\}$
$\epsilon \in]0,1]$, $(y_0^j)_{1\leqslant j\leqslant J} \in \mathbb{R}^M$, $x_0 = \widetilde{x} - \sum_{j=1}^{J} A_j^\top y_0^j$.

Main loop:
For $n = 0, 1, \ldots$

$$\left\lfloor\begin{array}{l}\gamma_n \in [\epsilon, 2-\epsilon] \\ j_n \in \{1,\ldots,J\} \\ \widetilde{y}_n^{j_n} = y_n^{j_n} + \gamma_n B_{j_n}^{-1} A_{j_n} x_n \\ y_{n+1}^{j_n} = \widetilde{y}_n^{j_n} - \gamma_n B_{j_n}^{-1} \text{prox}_{\gamma_n B_{j_n}^{-1}, h_{j_n}}\left(\gamma_n^{-1} B_{j_n}\widetilde{y}_n^{j_n}\right) \\ y_{n+1}^j = y_n^j, \quad \forall j \in \{1,\ldots,J\}\setminus\{j_n\} \\ x_{n+1} = x_n - A_{j_n}^\top (y_{n+1}^{j_n} - y_n^{j_n}).\end{array}\right.$$

Algorithm 1 benefits from the acceleration provided by variable metric methods through the introduction of preconditioning matrices $(B_j)_{1\leqslant j\leqslant J}$. Note that a non-preconditioned version is obtained by setting $\forall j \in \{1,\ldots,J\}\;\; B_j = \|A_j\|^2 I_{M_j}$. Moreover, when all the dual variables $y_n^{j_n}$ are updated in a parallel way followed by an update of the primal variable x_n, one recovers the Parallel Dual Forward-Backward proposed in [27].

Results in terms of convergence speed reveal the effectiveness of the above algorithm compared to existing algorithms in the literature and convergence guaranties on both generated primal and dual sequences are supplied.

III. Distributed algorithm

In order to generate a distributed or decentralized solution to Problem (5), we will make use of a global consensus technique by rewriting the problem under the following form:

$$\text{Find }\; \widehat{\boldsymbol{x}} = \underset{\boldsymbol{x}=(x^j)_{1\leqslant j\leqslant J}\in\Lambda_J}{\text{argmin}} \;\; \sum_{j=1}^{J} h_j(A_j x^j) + \frac{1}{2}\sum_{j=1}^{J}\omega_j\|x^j - \widetilde{x}\|^2, \tag{7}$$

where $(\omega_j)_{1\leqslant j\leqslant J} \in]0,1]^J$ are such that $\sum_{j=1}^{J}\omega_j = 1$, and Λ_J is the vector subspace of $\mathbb{R}^{NJ}$ that enforces the convergence toward the aggregate solution, defined as

$$\Lambda_J = \left\{[(x^1)^\top, \ldots, (x^J)^\top]^\top \in \mathbb{R}^{NJ} \;\middle|\; x^1 = \ldots = x^J\right\}. \tag{8}$$

One can notice that the solution to Problem (7) is linked to the solution to Problem (5) when the variables $(x^j)_{1\leqslant j\leqslant J}$ are all equal to $\widehat{x}$, that is the value of the proximity operator of $\sum_{j=1}^{J} h_j \circ A_j$ at $\widetilde{x}$ (see (5)).

A. Local Form of Consensus

For more flexibility, let us split the constraint set Λ_J into L local constraints Λ_{κ_ℓ} with cardinalities $(\kappa_\ell)_{1\leqslant\ell\leqslant L}$. Each constraint set Λ_{κ_ℓ} handles a subset $(\mathbb{V}_\ell)_{1\leqslant\ell\leqslant L}$ of $\{1,\ldots,J\}$ such that, for every $\boldsymbol{x} = [(x^1)^\top, \ldots, (x^J)^\top]^\top \in \mathbb{R}^{NJ}$,

$$\boldsymbol{x} \in \Lambda_J \quad\Leftrightarrow\quad (\forall \ell \in \{1,\ldots,L\}) \quad (x^j)_{j\in\mathbb{V}_\ell} \in \Lambda_{\kappa_\ell}. \tag{9}$$

Fig. 1 represents an example of a connected hypergraph induced by the sets $(\mathbb{V}_\ell)_{1\leqslant\ell\leqslant L}$. This hypergraph is composed of $J = 7$ nodes associated to the functions $(h_j)_{1\leqslant j\leqslant 7}$ and $L = 4$ hyperedges represented by the sets $(\mathbb{V}_\ell)_{1\leqslant\ell\leqslant 4}$ with cardinalities $\kappa_1 = 3, \kappa_2 = 2, \kappa_3 = 2$ and $\kappa_4 = 3$ respectively. Each function h_j is considered as local and processes its own private data. Moreover, each node j is allowed to communicate with nodes that belong to the same set $\mathbb{V}_\ell$.
For instance, node 4 belongs to the set $\mathbb{V}_2$ and communicates with node 5. Besides, node 3 belongs to $\mathbb{V}_1$ and $\mathbb{V}_4$ hence it is allowed to communicate with the nodes $\{1,2,5,7\}$. Note that the connectivity of the hypergraph is essential in order to ensure the convergence toward the consensus solution $\widehat{x}$.

Let us define, for every $\ell \in \{1,\ldots,L\}$, the matrix $\boldsymbol{S}_\ell \in \mathbb{R}^{N\kappa_\ell\times NJ}$ associated to a constraint Λ_{κ_ℓ} that extracts the vector $(x^j)_{j\in\mathbb{V}_\ell} \in \Lambda_{\kappa_\ell}$ from the concatenated vector $\boldsymbol{x} = [(x^1)^\top, \ldots, (x^J)^\top]^\top \in \mathbb{R}^{NJ}$:

$$(x^j)_{j\in\mathbb{V}_\ell} = [(x^{\mathrm{i}(\ell,1)})^\top, \ldots, (x^{\mathrm{i}(\ell,\kappa_\ell)})^\top]^\top = \boldsymbol{S}_\ell\,\boldsymbol{x}, \tag{10}$$

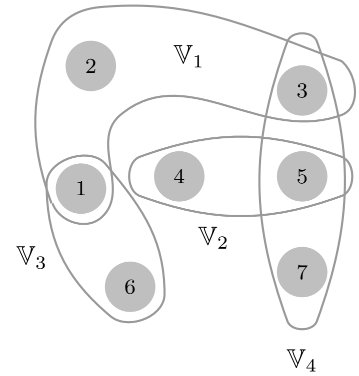

Fig. 1. Connected hypergraph of 7 nodes and 4 hyperedges.

where $\mathsf{i}(\ell,1),\ldots,\mathsf{i}(\ell,\kappa_\ell)$ denote the elements of $\mathbb{V}_\ell$ ordered in an increasing manner. The transpose matrix of $(\boldsymbol{S}_\ell)_{1\leqslant\ell\leqslant L}$ is such that, for every $v^\ell=(v^{\ell,k})_{1\leqslant k\leqslant\kappa_\ell}\in\mathbb{R}^{N\kappa_\ell}$,

$$\boldsymbol{x}=[(x^1)^\top,\ldots,(x^J)^\top]=\boldsymbol{S}_\ell^\top v^\ell, \tag{11}$$

where

$$x^j=\begin{cases} v^{\ell,k} & \text{if } j=\mathsf{i}(\ell,k) \text{ with } k\in\{1,\ldots,\kappa_\ell\}\\ 0 & \text{otherwise.}\end{cases} \tag{12}$$

The above definitions allow us to propose the following alternative formulation of Problem (7):

$$\text{Find }\ \widehat{\boldsymbol{x}}=\underset{\boldsymbol{x}=(x_j)_{1\leqslant j\leqslant J}\in\mathbb{R}^{NJ}}{\operatorname{argmin}}\ \sum_{j=1}^J h_j(A_jx_j)+\sum_{\ell=1}^L\iota_{\Lambda_{\kappa_\ell}}(\boldsymbol{S}_\ell\,\boldsymbol{x}) + \frac{1}{2}\sum_{j=1}^J\omega_j\|x_j-\widetilde{x}\|^2. \tag{13}$$

The main difference between formulations (7) and (13) is the introduction of the term $\sum_{\ell=1}^L\iota_{\Lambda_{\kappa_\ell}}(\boldsymbol{S}_\ell\,\boldsymbol{x})$ that allows updates in an asynchronous fashion and ensures the convergence to the aggregate solution.

In order to solve Problem (13) using Algorithm 1, it is necessary to set the following parameters:

- $J'=J+L$,
- $(\forall\ell\in\{1,\ldots,L\})\quad M_{J+\ell}=N\kappa_\ell$,
- $M=\sum_{j=1}^{J'}M_j$,
- $(\forall j\in\{1,\ldots,J\})\quad \boldsymbol{A}_j=[\ \underbrace{0\ldots0}_{N(j-1)\times}\ \omega_j^{-1/2}A_j\ \underbrace{0\ldots0}_{N(J-j)\times}\ ]$,
- $\boldsymbol{D}=\mathrm{Diag}\,(\omega_1^{-1/2}I_N,\ldots,\omega_J^{-1/2}I_N)$,
- $(\forall\ell\in\{1,\ldots,L\})\ h_{J+\ell}=\iota_{\Lambda_{\kappa_\ell}}$ and $\boldsymbol{A}_{J+\ell}=\boldsymbol{S}_\ell\boldsymbol{D}$.

Then, Problem (13) is recast as:

$$\text{Find }\ \widehat{\boldsymbol{x}}=\boldsymbol{D}\widehat{\boldsymbol{x}}'\quad\text{such that}$$

$$\widehat{\boldsymbol{x}}'=\underset{\boldsymbol{x}'\in\mathbb{R}^{NJ}}{\operatorname{argmin}}\ \sum_{j=1}^{J'}h_j(\boldsymbol{A}_j\boldsymbol{x}')+\frac{1}{2}\|\boldsymbol{x}'-\widetilde{\boldsymbol{x}}'\|^2, \tag{14}$$

where $\widetilde{\boldsymbol{x}}'=[\omega_1^{1/2}\widetilde{x}^\top,\ldots,\omega_J^{1/2}\widetilde{x}^\top]^\top\in\mathbb{R}^{NJ}$.

B. Proposed Algorithm

Using Algorithm 1 to solve Problem (14) leads to the following distributed algorithm after some reindexing and simplifications :

Algorithm 2 Distributed Preconditioned Dual Forward-Backward

Initialization:

$(\omega_j)_{1\leqslant j\leqslant J}\in]0,1]^J$ such that $\sum_{j=1}^J\omega_j=1$

$B_j\in\mathbb{R}^{M_j\times M_j}$ with $B_j\succeq A_jA_j^\top,\quad j\in\{1,\ldots,J\}$

$\vartheta_\ell=\min_{j\in\mathbb{V}_\ell}\omega_j,\quad \ell\in\{1,\ldots,L\}$

$\epsilon\in]0,1]$

$y_0^j\in\mathbb{R}^{M_j}, x_0^j=\widetilde{x}-A_j^\top y_0^j,\quad j\in\{1,\ldots,J\}.$

Main loop:

For $n=0,1,\ldots$

$\gamma_n\in[\epsilon,2-\epsilon]$

$j_n\in\{1,\ldots,J+L\}$

If $j_n\leqslant J$

Local optimization:

$\widetilde{y}_n^{j_n}=y_n^{j_n}+\gamma_nB_{j_n}^{-1}A_{j_n}x_n^{j_n}$

$y_{n+1}^{j_n}=\widetilde{y}_n^{j_n}-\gamma_nB_{j_n}^{-1}\mathrm{prox}_{\gamma_n\omega_{j_n}B_{j_n}^{-1},h_{j_n}}\left(\gamma_n^{-1}B_{j_n}\widetilde{y}_n^{j_n}\right)$

$y_{n+1}^j=y_n^j,\quad j\in\{1,\ldots,J\}\setminus\{j_n\}$

$x_{n+1}^{j_n}=x_n^{j_n}-A_{j_n}^\top(y_{n+1}^{j_n}-y_n^{j_n})$

$x_{n+1}^j=x_n^j,\quad j\in\{1,\ldots,J\}\setminus\{j_n\}$

else

Synchronization:

$\ell_n=j_n-J$

$y_{n+1}^j=y_n^j,\quad j\in\{1,\ldots,J\}$

$\overline{x}_n^{\ell_n}=\dfrac{1}{\kappa_{\ell_n}}\sum_{j\in\mathbb{V}_{\ell_n}}x_n^j$

For $k=1,\ldots,\kappa_{\ell_n}$

$x_{n+1}^{\mathsf{i}(\ell_n,k)}=x_n^{\mathsf{i}(\ell_n,k)}+\gamma_n\vartheta_{\ell_n}\omega_{\mathsf{i}(\ell_n,k)}^{-1}(\overline{x}_n^{\ell_n}-x_n^{\mathsf{i}(\ell_n,k)})$

$x_{n+1}^j=x_n^j,\quad j\notin\mathbb{V}_{\ell_n}.$

One can notice that Algorithm 2 is composed of two main parts:

- First a local optimization part which is reminiscent of the Dual Block Forward-Backward algorithm where, at each iteration, a block j_n is selected and the associated dual and primal variables $y_n^{j_n}$ and $x_n^{j_n}$ respectively are updated. Note that the difference between the proposed algorithm and Algorithm 1 lies in the fact that each block j_n is now associated to a primal variable $x_n^{j_n}$ whereas, in Algorithm 1 x_n was a shared variable.
- The second part of Algorithm 2 is a synchronization step in-which a set $\mathbb{V}_{\ell_n}$ is selected and all the variables $(x^{j_n})_{j_n \in \mathbb{V}_{\ell_n}}$ are updated by computing the average over the selected set $\mathbb{V}_{\ell_n}$.

In Algorithm 2 all computation steps only involve local variables, so considerable flexibility is allowed by the quasi-cyclic rule for choosing the indices j_n and ℓ_n at each iteration n. This distributed algorithm inherits all the advantages of primal-dual methods, in particular it requires no inversion of the matrices $(A_j)_{1 \leqslant j \leqslant J}$, which is of main interest when these matrices do not have a simple structure and are of very large size. Note that the proposed approach is quite different from the ones developed in [20] since it does not implement a random sweeping rule and the convergence results do not rely on nonexpansiveness properties of some stochastic operators.

Remark that the case of a graph topology is encompassed by Algorithm 2, by setting for every $\ell \in \{1, \ldots, L\}$ the cardinality of the sets $\mathbb{V}_\ell$ to $\kappa_\ell = 2$, and requiring after each optimization step on block j_n, a synchronization step on all the sets $\mathbb{V}_\ell$ to which j_n belongs.

C. *Special Case when L = 1*

An interesting instance of Algorithm 2 is obtained by setting $L = 1$, $\mathbb{V}_1 = \{1, \ldots, J\}$, and by performing at each iteration an update of a subset of the dual variables followed by a global averaging step. After some reindexing of the involved variables, this yields Algorithm 3.

This algorithm allows to solve Problem (7) by computing the dual variables in a parallel manner. At iteration $n \in \mathbb{N}$, it allows to activate only a subset $(y_n^j)_{j \in \mathbb{J}_n}$ of them. It should be emphasized that even in the case when all the dual variables are updated iteratively (i.e. $(\forall n \in \mathbb{N})\ \mathbb{J}_n = \{1, \ldots, J\}$), Algorithm 3 exhibits a different structure from the Parallel Dual Forward-Backward Algorithm in [27].

IV. Conclusion

We have proposed a new asynchronous primal-dual algorithm for computing the proximity operator of a sum of convex functions composed with arbitrary linear operators, which is a frequently encountered problem in various application fields. The proposed algorithm benefits from the flexibility offered by variable metric techniques that can significantly improve its convergence speed. The convergence properties of this algorithms will be discussed in a forthcoming paper.
As future work, we also intend to apply the proposed algorithm to large-scale inverse problems, especially those encountered in video restoration [28].

Algorithm 3 Parallel Preconditioned Dual Forward-Backward

Initialization:

$(\omega_j)_{1 \leqslant j \leqslant J} \in]0,1]^J$ such that $\displaystyle\sum_{j=1}^{J} \omega_j = 1$

$B_j \in \mathbb{R}^{M_j \times M_j}$ with $B_j \succeq A_j A_j^\top, \quad j \in \{1, \ldots, J\}$

$\vartheta = \min_{j \in \{1, \ldots, J\}} \omega_j$

$\epsilon \in]0,1]$

$y_0^j \in \mathbb{R}^{M_j}, x_0^j = \widetilde{x} - A_j^\top y_0^j, \quad j \in \{1, \ldots, J\}.$

Main loop:

For $n = 0, 1, \ldots$

$$
\left\lfloor
\begin{array}{l}
\gamma_n \in [\epsilon, 2-\epsilon] \\
\mathbb{J}_n \subset \{1, \ldots, J\} \\
\text{For } j \in \mathbb{J}_n \\
\left\lfloor
\begin{array}{l}
\widetilde{y}_n^j = y_n^j + \gamma_n B_j^{-1} A_j x_n^j \\
y_{n+1}^j = \widetilde{y}_n^j - \gamma_n B_j^{-1} \mathrm{prox}_{\gamma_n \omega_j B_j^{-1}, h_j}\big(\gamma_n^{-1} B_j \widetilde{y}_n^j\big) \\
x_{n+1/2}^j = x_n^j - A_j^\top (y_{n+1}^j - y_n^j)
\end{array}
\right. \\
\text{For } j \in \{1, \ldots, J\} \setminus \mathbb{J}_n \\
\left\lfloor
\begin{array}{l}
y_{n+1}^j = y_n^j \\
x_{n+1/2}^j = x_n^j
\end{array}
\right. \\
\overline{x}_n = \dfrac{1}{J} \displaystyle\sum_{j=1}^{J} x_{n+1/2}^j \\
\text{For } j = 1, \ldots, J \\
\left\lfloor
\begin{array}{l}
x_{n+1}^j = x_{n+1/2}^j + \gamma_n \vartheta \omega_j^{-1} (\overline{x}_n - x_{n+1/2}^j).
\end{array}
\right.
\end{array}
\right.
$$

References

[1] P. L. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal processing," in *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, H. H. Bauschke, R. Burachik, P. L. Combettes, V. Elser, D. R. Luke, and H. Wolkowicz, Eds. New York: Springer-Verlag, 2010, pp. 185–212.

[2] N. Parikh and S. Boyd, "Proximal algorithms," *Found. Trends Optim.*, vol. 1, no. 3, pp. 127–239, Jan. 2014.

[3] N. Komodakis and J.-C. Pesquet, "Playing with duality: An overview of recent primal-dual approaches for solving large-scale optimization problems," *IEEE Signal Process. Mag.*, vol. 32, no. 6, pp. 31–54, Oct. 2015.

[4] P. L. Combettes, D. Dung, and B. C. Vũ, "Dualization of signal recovery problems," *Set-Valued Var. Anal.*, vol. 18, no. 3, pp. 373–404, Dec. 2010.

[5] L. Condat, "A primal-dual splitting method for convex optimization involving lipschitzian, proximable and linear composite terms," *J. Optim. Theory App.*, vol. 158, no. 2, pp. 460–479, Aug. 2013.

[6] S. R. Becker and P. L. Combettes, "An algorithm for splitting parallel sums of linearly composed monotone operators, with applications to signal recovery," *J. Nonlinear Convex Anal.*, vol. 15, no. 1, pp. 137–159, Jan. 2014.

[7] C. Couprie, L. Grady, L. Najman, J.-C. Pesquet, and H. Talbot, "Dual constrained tv-based regularization on graphs," *SIAM J. Imaging Sci.*, vol. 6, pp. 1246–1273, Jul. 2013.

[8] A. Jezierska, E. Chouzenoux, J.-C. Pesquet, and H. Talbot, "A primal-dual proximal splitting approach for restoring data corrupted with poisson-gaussian noise," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2012)*, Kyoto, Japan, 25-30 Mar. 2012, pp. 1085–1088.

[9] A. Chambolle and T. Pock, "A first-order primal-dual algorithm for convex problems with applications to imaging," *J. Math. Imaging Vision*, vol. 40, no. 1, pp. 120–145, May 2011.

[10] P. L. Combettes, L. Condat, J.-C. Pesquet, and B. C. Vũ, "A forward-backward view of some primal-dual optimization methods in image recovery," in *IEEE International Conference on Image Processing (ICIP 2014)*, Paris, France, 27-30 Oct. 2014, pp. 4141–4145.

[11] R. I. Boţ and C. Hendrich, "A Douglas-Rachford type primal-dual method for solving inclusions with mixtures of composite and parallel-sum type monotone operators," *SIAM J. Optim.*, vol. 23, no. 4, pp. 2541–2565, Dec. 2013.

[12] P. L. Combettes and J.-C. Pesquet, "Stochastic quasi-Fejér block-coordinate fixed point iterations with random sweeping," *SIAM J. Optim.*, vol. 25, no. 2, pp. 1221–1248, Jul. 2015.

[13] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Machine Learn.*, vol. 8, no. 1, pp. 1–122, Jan. 2011.

[14] F. Iutzeler, P. Bianchi, P. Ciblat, and W. Hachem, "Explicit convergence rate of a distributed alternating direction method of multipliers," *to appear in IEEE Trans. Autom. Control*, Dec. 2014.

[15] S. Shalev-Shwartz and T. Zhang, "Stochastic dual coordinate ascent methods for regularized loss minimization," *J. Mach. Learn. Res.*, vol. 14, no. 1, pp. 567–599, Feb. 2013.

[16] A. Chambolle and T. Pock, "A remark on accelerated block coordinate descent for computing the proximity operators of a sum of convex functions," Tech. Rep., 2015, http://www.optimization-online.org/DB_FILE/2015/01/4719.pdf.

[17] P. Bianchi, W. Hachem, and F. Iutzeler, "A stochastic coordinate descent primal-dual algorithm and applications to large-scale composite optimization," 2014, http://arxiv.org/pdf/1407.0898v3.pdf.

[18] Z. Qu, P. Richtárik, and T. Zhang, "Randomized dual coordinate ascent with arbitrary sampling," 2014, http://arxiv.org/abs/1411.5873.

[19] M. Jaggi, V. Smith, M. Takac, J. Terhorst, S. Krishnan, T. Hofmann, and M. I. Jordan, "Communication-efficient distributed dual coordinate ascent," in *Adv. Neural Inf. Process. Syst.*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, Eds. Curran Associates, Inc., Sept. 2014, pp. 3068–3076.

[20] J.-C. Pesquet and A. Repetti, "A class of randomized primal-dual algorithms for distributed optimization," *J. Nonlinear Convex Anal.*, vol. 16, no. 12, Dec. 2015.

[21] P. Richtárik and M. Takáĉ, "Distributed coordinate descent method for learning with big data," 2013, http://arxiv.org/pdf/1310.2059v1.pdf.

[22] F. Abboud, E. Chouzenoux, J.-C. Pesquet, J.-H. Chenot, and L. Laborelli, "A dual block coordinate proximal algorithm with application to deconvolution of interlaced video sequences," in *IEEE International Conference on Image Processing (ICIP 2015)*, Quebec City, Canada, 27-30 Sep. 2015, 5 pages.

[23] E. Chouzenoux, J.-C. Pesquet, and A. Repetti, "Variable metric forward-backward algorithm for minimizing the sum of a differentiable function and a convex function," *J. Optim. Theory App.*, vol. 162, no. 1, pp. 107–132, Jul. 2014.

[24] E. Chouzenoux, J.-C. Pesquet, and A. Repetti, "A block coordinate variable metric forward-backward algorithm," Tech. Rep., 2013, http://www.optimization-online.org/DB_HTML/2013/12/4178.html.

[25] S. Becker and J. Fadili, "A quasi-Newton proximal splitting method," in *Adv. Neural Inf. Process. Syst.*, Lake Tahoe, Nevada, 3-8 Dec. 2012, pp. 2627–2635.

[26] P. L. Combettes and B. C. Vũ, "Variable metric forward-backward splitting with applications to monotone inclusions in duality," *Optimization*, vol. 63, no. 9, pp. 1289–1318, Sept. 2014.

[27] P. L. Combettes, D. Dũng, and B. C. Vũ, "Proximity for sums of composite functions," *J. Math. Anal. Appl.*, vol. 380, no. 2, pp. 680–688, Aug. 2011.

[28] F. Abboud, E. Chouzenoux, J.-C. Pesquet, J.-H. Chenot, and L. Laborelli, "A hybrid alternating proximal method for blind video restoration," in *22nd IEEE European Signal Processing Conference (EUSIPCO 2014)*, Lisbon, Portugal, 1-5 Sep. 2014, pp. 1811–1815.