Noname manuscript No.
(will be inserted by the editor)

Combining Lagrangian Decomposition and Excessive Gap Smoothing Technique for Solving Large-Scale Separable Convex Optimization Problems

Tran Dinh Quoc · **Carlo Savorgnan** · **Moritz Diehl**

Received: date / Accepted: date

Abstract A new algorithm for solving large-scale convex optimization problems with a separable objective function is proposed. The basic idea is to combine three techniques: Lagrangian dual decomposition, excessive gap and smoothing. The main advantage of this algorithm is that it dynamically updates the smoothness parameters which leads to numerically robust performance. The convergence of the algorithm is proved under weak conditions imposed on the original problem. The rate of convergence is $O(\frac{1}{k})$, where k is the iteration counter. In the second part of the paper, the algorithm is coupled with a dual scheme to construct a switching variant of the dual decomposition. We discuss implementation issues and make a theoretical comparison. Numerical examples confirm the theoretical results.

Keywords Excessive gap · smoothing technique · Lagrangian decomposition · proximal mappings · large-scale problem · separable convex optimization · distributed optimization.

1 Introduction

Large-scale convex optimization problems appear in many areas of science such as graph theory, networks, transportation, distributed model predictive control, distributed estimation and multistage stochastic optimization [8,17,21,22,24,32,34,38,39,40,41]. Solving large-scale optimization problems is still a challenge in many applications [9]. Over the years, thanks to the development of parallel and distributed computer systems, the chances for solving large-scale problems have been increased. However, methods and algorithms for solving this type of problems are limited [2,9].

Convex minimization problems with a separable objective function form a class of problems which is relevant in many applications. This class of problems is also known as separable convex minimization problems, see, e.g. [2]. Without loss of generality, a separable convex optimization problem can be written in the form of a convex program with separable objective function and coupled linear constraints [2]. In addition, decoupling convex

Tran Dinh Quoc · Carlo Savorgnan · Moritz Diehl
Department of Electrical Engineering (ESAT-SCD) and Optimization in Engineering Center (OPTEC), K.U. Leuven, Kasteelpark Arenberg 10, B-3001 Leuven, Belgium.
E-mail: {quoc.trandinh, carlo.savorgnan, moritz.diehl}@esat.kuleuven.be
Tran Dinh Quoc, Hanoi University of Science, Hanoi, Vietnam.

constraints may also be considered. Mathematically, this problem can be formulated in the following form:

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \ & \phi(x) := \sum_{i=1}^{M} \phi_i(x_i) \\ \text{s.t.} \ & x_i \in X_i \ (i = 1,\cdots,M), \\ & \sum_{i=1}^{M} A_i x_i = b, \end{aligned} \tag{1}$$

where $\phi_i : \mathbb{R}^{n_i} \to \mathbb{R}$ is convex, $X_i \in \mathbb{R}^{n_i}$ is a nonempty, closed convex set, $A_i \in \mathbb{R}^{m\times n_i}$, $b \in \mathbb{R}^m$ for all $i = 1,\ldots,M$, and $n_1 + n_2 + \cdots + n_M = n$. The last constraint is called *coupling linear constraint*. In principle, many convex problems can be written in this separable form by doubling the variables, i.e. introducing new variables x_i and imposing the constraint $x_i = x$. Despite the increased number of variables, treating convex problems by doubling variables may be useful in some situations, see, e.g. [11,12].

In the literature, numerous approaches have been proposed for solving problem (1). For example, (augmented) Lagrangian relaxation and subgradient methods of multipliers [2, 13,33,39], Fenchel's dual decomposition [15], alternating linearization [6,12,23], proximal point-type methods [4,7,37], interior point methods [21,41,25,36], mean value cross decomposition [18] and partial inverse method [35] among many others have been proposed. Our motivation in this paper is to develop a numerical algorithm for solving (1) which can be implemented in a parallel or distributed fashion. Note that the approach presented in the present paper is different from splitting methods and alternating methods considered in the literature, see, e.g. [6,10].

One of the classical approaches for solving (1) is Lagrangian dual decomposition. The main idea of this approach is to solve the dual problem by means of a subgradient method. It has been recognized in practice that subgradient methods are usually slow and numerically sensitive to the step size parameters. In the special case of a strongly convex objective function, the dual function is differentiable. Consequently, gradient schemes can be applied to solve the dual problem.

Recently, Nesterov [29] developed smoothing techniques for solving nonsmooth convex optimization problems based on the fast gradient scheme which was introduced in his early work [28]. The fast gradient schemes have been used in numerous applications including image processing, compressed sensing, networks and system identification [1,5,14,16,12, 26].

Exploiting Nesterov's idea in [30], Necoara and Suykens [27] applied a smoothing technique to the dual problem in the framework of Lagrangian dual decomposition and then used the fast gradient scheme to maximize the smoothed function of the dual problem. This resulted in a new variant of dual decomposition algorithms for solving separable convex optimization. The authors proved that the rate of convergence of their algorithm is $O(\frac{1}{k})$ which is much better than $O(\frac{1}{\sqrt{k}})$ in the subgradient methods of multipliers, where k is the iteration counter. A main disadvantage of this scheme is that the smoothness parameter requires to be given *a priori*. Moreover, this parameter crucially depends on the given desired accuracy. Since the Lipschitz constant of the gradient of the objective function in the dual problem is inversely proportional to the smoothness parameter, the algorithm usually generates short steps towards a solution of the problem although the rate of convergence is $O(\frac{1}{k})$.

To overcome this drawback, in this paper, we propose a new algorithm which combines three techniques: smoothing [30,31], excessive gap [31] and Lagrangian dual decomposition [2] techniques. Instead of fixing the smoothness parameters, we update them dynamically

at every iteration. Even though the worst case complexity is $O(\frac{1}{\varepsilon})$, where ε is a given tolerance, the algorithms developed in this paper work better than the one in [27] and are more numerically robust in practice. Note that the computational cost of the proposed algorithms remains almost the same as in the proximal-center-based decomposition algorithm proposed in [27, Algorithm 3.2]. (Algorithm 3.2 in [27] requires to compute an additional dual step). This algorithm is called dual decomposition with primal update (Algorithm 1). Alternatively, we apply the switching strategy of [31] to obtain a decomposition algorithm with switching primal-dual update for solving problem (1). This algorithm differs from the one in [31] at two points. First, the smoothness parameter is dynamically updated with an exact formula and second the proximal-based mappings are used to handle the nonsmoothness of the objective function. The second point is more significant since, in practice, estimating the Lipschitz constants is not an easy task even if the objective function is differentiable. The switching algorithm balances the disadvantage of the decomposition methods using the primal update (Algorithm 1) and the dual update (Algorithm 3.2 [27]). Proximal-based mapping only plays a role of handling the nonsmoothness of the objective function. Therefore, the algorithms developed in this paper do not belong to any proximal-point algorithm class considered in the literature. Note also that all algorithms developed in this paper are first order methods which can be highly distributed.

Contribution. The contribution of this paper is the following:

1. We apply the Lagrangian relaxation, smoothing and excessive gap techniques to large-scale separable convex optimization problems which are not necessarily smooth. Note that the excessive gap condition that we use in this paper is different from the one in [31], where not only the duality gap is measured but also the feasibility gap is used in the framework of constrained optimization, see (23).
2. We propose two algorithms for solving general separable convex optimization problems. The first algorithm is new, while the second one is a new variant of the first algorithm proposed in [31, Algorithm 1] applied to Lagrangian dual decomposition. A special case of the algorithms, where the objective is strongly convex is considered. All the algorithms are highly parallelizable and distributed.
3. The convergence of the algorithms is proved and the rate of convergence is estimated. Implementation details are discussed and a theoretical and numerical comparison is made.

The rest of this paper is organized as follows. In the next section, we briefly describe the Lagrangian dual decomposition method [2] for separable convex optimization, the smoothing technique via prox-functions as well as excessive gap techniques [31]. We also provide several technical lemmas which will be used in the sequel. Section 3 presents a new algorithm called *decomposition algorithm with primal update* and estimates its worst-case complexity. Section 4 is a combination of the primal and the dual step update schemes which is called *decomposition algorithm with primal-dual update*. Section 5 is an application of the dual scheme (55) to the strongly convex case of problem (2). We also discuss the implementation issues of the proposed algorithms and a theoretical comparison of Algorithms 1 and 2 in Section 6. Numerical examples are presented in Section 7 to examine the performance of the proposed algorithms and to compare different methods.

Notation. Throughout the paper, we shall consider the Euclidean space $\mathbb{R}^n$ endowed with an inner product $x^T y$ for $x, y \in \mathbb{R}^n$ and the norm $\|x\| := \sqrt{x^T x}$. Associated with $\|\cdot\|$, $\|\cdot\|_* := \max\{(\cdot)^T x : \|x\| \leq 1\}$ defines its dual norm. For simplicity of discussion, we use the Euclidean norm in the whole paper. Hence, $\|\cdot\|_*$ is equivalent to $\|\cdot\|$. The notation

$x=(x_1,\ldots,x_M)$ represents a column vector in $\mathbb{R}^n$, where x_i is a subvector in $\mathbb{R}^{n_i}$, $i=1,\ldots,M$ and $n_1+\cdots+n_M=n$.

2 Lagrangian dual decomposition and excessive gap smoothing technique

A classical technique to address coupling constraints in optimization is Lagrangian relaxation [2]. However, this technique often leads to a nonsmooth optimization problem in the dual form. To overcome this situation, we combine the Lagrangian dual decomposition and smoothing technique in [30,31] to obtain a smoothly approximate dual problem.

For simplicity of discussion, we consider problem (1) with $M=2$. However, the methods presented in the next sections can be directly applied to the case $M>2$ (see Section 6). The problem (1) can be rewritten as follows:

$$\phi^* := \left\{ \begin{array}{ll} \min\limits_{x:=(x_1,x_2)} & \phi(x) := \phi_1(x_1)+\phi_2(x_2) \\ \text{s.t.} & A_1x_1+A_2x_2=b \\ & x\in X_1\times X_2 := X, \end{array} \right. \tag{2}$$

where $\phi_i:\mathbb{R}^{n_i}\to\mathbb{R}$ is convex, X_i is a nonempty, closed, convex and bounded subset in $\mathbb{R}^{n_i}$, $A_i\in\mathbb{R}^{m\times n_i}$ and $b\in\mathbb{R}^m$ $(i=1,2)$. Problem (2) is said to satisfy the Slater constraint qualification condition if $\mathrm{ri}(X)\cap\{x=(x_1,x_2)\mid A_1x_1+A_2x_2=b\}\neq\emptyset$, where $\mathrm{ri}(X)$ is the relative interior of the convex set X. Let us denote by X^* the solution set of this problem. Throughout the paper, we assume that:

A.1 *The solution set X^* is nonempty and either the Slater qualification condition for problem* (2) *holds or X_i is polyhedral. The function ϕ_i is proper, lower semicontinuous and convex in $\mathbb{R}^n$, $i=1,2$.*

Since X is convex and bounded, X^* is also convex and bounded. Note that the objective function ϕ is not necessarily smooth. For example, $\phi(x)=\|x\|_1=\sum_{i=1}^n|x_{(i)}|$, which is is nonsmooth and separable.

2.1 Decomposition via Lagrangian relaxation

Let us define the Lagrange function of the problem (2) with respect to the coupling constraint $A_1x_1+A_2x_2=b$ as:

$$L(x,y) := \phi_1(x_1)+\phi_2(x_2)+y^T(A_1x_1+A_2x_2-b), \tag{3}$$

where $y\in\mathbb{R}^m$ is the multiplier associated with the coupling constraint $A_1x_1+A_2x_2=b$. A triplet $(x_1^*,x_2^*,y^*)\in X\times\mathbb{R}^m$ is called a saddle point of L if:

$$L(x^*,y)\le L(x^*,y^*)\le L(x,y^*),\ \forall x\in X,\ \forall y\in\mathbb{R}^m. \tag{4}$$

Next, we define the Lagrange dual function d of the problem (2) as:

$$d(y) := \min_{x\in X}\left\{L(x,y) := \phi_1(x_1)+\phi_2(x_2)+y^T(A_1x_1+A_2x_2-b)\right\}. \tag{5}$$

and then write down the dual problem of (2):

$$d^* := \max_{y\in\mathbb{R}^m} d(y). \tag{6}$$

Let $A = [A_1, A_2]$. Due to Assumption A.1 *strong duality* holds and we have:

$$d^* = \max_{y \in \mathbb{R}^m} d(y) \overset{\text{strong duality}}{=} \min_{x \in X} \{\phi(x) \mid Ax = b\} = \phi^*. \tag{7}$$

Let us denote by Y^* the solution set of the dual problem (6). It is well-known that Y^* is bounded due to Assumption A.1.

Now, let us consider the dual function d defined by (5). It is important to note that the dual function $d(y)$ can be computed separately as:

$$d(y) = d_1(y) + d_2(y) - b^T y, \tag{8}$$

where

$$d_i(y) := \min_{x_i \in X_i} \left\{ \phi_i(x_i) + y^T A_i x_i \right\}, \ i = 1, 2. \tag{9}$$

We denote by $x_i^*(y)$ a solution of the minimization problem in (9) ($i = 1, 2$) and $x^*(y) := (x_1^*(y), x_2^*(y))$. Since ϕ_i is continuous and X_i is closed and bounded, this problem has a solution. Note that if $x_i^*(y)$ is not uniques for a given y then d_i is not differentiable at the point y ($i = 1, 2$). Consequently, d is not differentiable at y. The representation (8)-(9) is called a *dual decomposition* of the dual function d.

2.2 Smoothing the dual function via prox-functions

By assumption that X_i is bounded, instead of considering the nonsmooth function d, we smooth the dual function d by means of prox-functions. A function p_i is called a proximity function (prox-function) of a given nonempty, closed and bounded convex set $X_i \subset \mathbb{R}^{n_i}$ if p_i is continuous, strongly convex with convexity parameter $\sigma_i > 0$ and $X_i \subseteq \mathrm{dom}(p_i)$.

Suppose that p_i is a prox-function of X_i and $\sigma_i > 0$ is its convexity parameter ($i = 1, 2$). Let us consider the following functions:

$$d_i(y; \beta_1) := \min_{x_i \in X_i} \left\{ \phi_i(x_i) + y^T A_i x_i + \beta_1 p_i(x_i) \right\}, \ i = 1, 2, \tag{10}$$

$$d(y; \beta_1) := d_1(y; \beta_1) + d_2(y; \beta_1) - b^T y. \tag{11}$$

Here, $\beta_1 > 0$ is a given parameter called smoothness parameter. We denote by $x_i^*(y; \beta_1)$ the solution of (10), i.e.:

$$x_i^*(y; \beta_1) := \operatorname*{argmin}_{x_i \in X_i} \left\{ \phi_i(x_i) + y^T A_i x_i + \beta_1 p_i(x_i) \right\}, \ i = 1, 2. \tag{12}$$

Note that it is possible to use different parameters β_1^i for (10) ($i = 1, 2$).

Let x_i^c be the prox-center of X_i which is defined as:

$$x_i^c = \operatorname*{argmin}_{x_i \in X_i} p_i(x_i), \ i = 1, 2. \tag{13}$$

Without loss of generality, we can assume that $p_i(x_i^c) = 0$. Since X_i is bounded, the quantity

$$D_i := \max_{x_i \in X_i} p_i(x_i) \tag{14}$$

is well-defined and $0 \leq D_i < +\infty$ for $i = 1, 2$. The following lemma shows the main properties of $d(\cdot; \beta_1)$, whose proof can be found, e.g., in [27, 31].

Lemma 1 *For any $\beta_1 > 0$, the function $d_i(\cdot;\beta_1)$ defined by* (10) *is well-defined and continuously differentiable on $\mathbb{R}^m$. Moreover, this function is concave and its gradient w.r.t y is given as:*

$$\nabla d_i(y;\beta_1) = A_i x_i^*(y;\beta_1),\ i = 1,2, \tag{15}$$

which is Lipschitz continuous with a Lipschitz constant $L_i^d(\beta_1) = \frac{\|A_i\|^2}{\beta_1\sigma_i}$ $(i = 1,2)$. The following estimates hold:

$$d_i(y;\beta_1) \geq d_i(y) \geq d_i(y;\beta_1) - \beta_1 D_i,\ i = 1,2. \tag{16}$$

Consequently, the function $d(\cdot;\beta_1)$ defined by (11) *is concave and differentiable and its gradient is given by $\nabla d(y;\beta_1) := Ax^*(y;\beta_1) - b$ which is Lipschitz continuous with a Lipschitz constant $L^d(\beta_1) := \frac{1}{\beta_1}\sum_{i=1}^2 \frac{\|A_i\|^2}{\sigma_i}$. Moreover, it holds that:*

$$d(y;\beta_1) \geq d(y) \geq d(y;\beta_1) - \beta_1(D_1 + D_2). \tag{17}$$

The inequalities (17) show that $d(\cdot;\beta_1)$ is an approximation of d. Moreover, $d(\cdot;\beta_1)$ converges to d as β_1 tends to zero.

Remark 1 Even without the assumption that X is bounded, if the solution set X^* of (2) is bounded then, in principle, we can bound the feasible set X by a large compact set which contains all the sampling points generated by the algorithms (see Section 4 below). However, in the following algorithms we do not use D_i, $i = 1,2$ (defined by (14)) in any computational step. They only appear in the theoretical complexity estimates.

Next, for a given $\beta_2 > 0$, we define a mapping $\psi(\cdot;\beta_2)$ from X to $\mathbb{R}$ by:

$$\psi(x;\beta_2) := \max_{y\in\mathbb{R}^m}\left\{(Ax-b)^T y - \frac{\beta_2}{2}\|y\|^2\right\}. \tag{18}$$

This function can be considered as an approximate version of $\psi(x) := \max_{y\in\mathbb{R}^m}\left\{(Ax-b)^T y\right\}$ using the prox-function $p(y) := \frac{1}{2}\|y\|^2$. It is easy to show that the unique solution of the maximization problem in (18) is given explicitly as $y^*(x;\beta_2) = \frac{1}{\beta_2}(Ax-b)$ and $\psi(x;\beta_2) = \frac{1}{2\beta_2}\|Ax-b\|^2$. Therefore, $\psi(\cdot;\beta_2)$ is well-defined and differentiable on X. Let

$$f(x;\beta_2) := \phi(x) + \psi(x;\beta_2) = \phi(x) + \frac{1}{2\beta_2}\|Ax-b\|^2. \tag{19}$$

The next lemma summarizes the properties of $\psi(\cdot;\beta_2)$.

Lemma 2 *For any $\beta_2 > 0$, the function $\psi(\cdot;\beta_2)$ defined by* (18) *is continuously differentiable on X and its gradient is given by:*

$$\nabla\psi(x;\beta_2) = (\nabla_{x_1}\psi(x;\beta_2), \nabla_{x_2}\psi(x;\beta_2)) = (A_1^T y^*(x;\beta_2),\ A_2^T y^*(x;\beta_2)), \tag{20}$$

which is Lipschitz continuous with a Lipschitz constant $L^\psi(\beta_2) := \frac{1}{\beta_2}(\|A_1\|^2 + \|A_2\|^2)$. Moreover, the following estimate holds for all $x, \hat{x} \in X$:

$$\begin{aligned}\psi(x;\beta_2) \ &\leq \psi(\hat{x};\beta_2) + \nabla_1\psi(\hat{x};\beta_2)^T(x_1-\hat{x}_1) + \nabla_2\psi(\hat{x};\beta_2)^T(x_2-\hat{x}_2)\\ &\quad + \frac{L_1^\psi(\beta_2)}{2}\|x_1-\hat{x}_1\|^2 + \frac{L_2^\psi(\beta_2)}{2}\|x_2-\hat{x}_2\|^2,\end{aligned} \tag{21}$$

where $L_1^\psi(\beta_2) := \frac{2}{\beta_2}\|A_1\|^2$ and $L_2^\psi(\beta_2) := \frac{2}{\beta_2}\|A_2\|^2$.

Proof Since $\psi(x;\beta_2) = \frac{1}{2\beta_2}\|A_1x_1 + A_2x_2 - b\|^2$ by the definition (18) and $y^*(x;\beta_2) = \frac{1}{\beta_2}(A_1x_1 + A_2x_2 - b)$, it is easy to compute directly $\nabla\psi(\cdot;\beta_2)$. Moreover, we have:

$$\begin{aligned}\psi(x;\beta_2) - \psi(\hat{x};\beta_2) - \nabla\psi(\hat{x};\beta_2)^T(x-\hat{x}) &= \frac{1}{2\beta_2}\|A_1(x_1-\hat{x}_1) + A_2(x_2-\hat{x}_2)\|^2 \\ &\leq \frac{1}{\beta_2}\|A_1\|^2\|x_1-\hat{x}_1\|^2 + \frac{1}{\beta_2}\|A_2\|^2\|x_2-\hat{x}_2\|^2.\end{aligned} \tag{22}$$

This inequality is indeed (21). □

From the definition of $f(\cdot;\beta_2)$, we obtain:

$$f(x;\beta_2) - \frac{1}{2\beta_2}\|Ax-b\|^2 = \phi(x) \leq f(x;\beta_2). \tag{23}$$

Note that $f(\cdot;\beta_2)$ is an upper bound of $\phi(\cdot)$ instead of a lower bound as in [31]. Note that the Lipschitz constants in (21) are roughly estimated. These quantities can be quantified carefully by taking into account the problem structure to trade-off the computational effort in each component subproblem.

2.3 Excessive gap technique

Since the primal-dual gap of the primal and dual problems (2)-(6) is measured by $g(x,y) := \phi(x) - d(y)$, if the gap g is equal to zero for some feasible point x and y then this point is an optimal solution of (2)-(6). In this section, we apply to the Lagrangian dual decomposition framework a technique called *excessive gap* proposed by Nesterov in [31].

Let us consider $\hat{d}(y;\beta_1) := d(y;\beta_1) - \beta_1(D_1 + D_2)$. It follows from (17) and (23) that $\hat{d}(\cdot;\beta_1)$ is an underestimate of $d(\cdot)$, while $f(\cdot;\beta_2)$ is an overestimate of $\phi(\cdot)$. Therefore, $0 \leq g(x,y) = \phi(x) - d(y) \leq f(x;\beta_2) - d(y;\beta_1) + \beta_1(D_1 + D_2)$. Let us recall the following excessive gap condition introduced in [31].

Definition 1 We say that a point $(\bar{x},\bar{y}) \in X \times \mathbb{R}^m$ satisfies the *excessive gap* condition with respect to two smoothness parameters $\beta_1 > 0$ and $\beta_2 > 0$ if:

$$f(\bar{x};\beta_2) \leq d(\bar{y};\beta_1), \tag{24}$$

where $f(\cdot;\beta_2)$ and $d(\cdot;\beta_1)$ are defined by (23) and (11), respectively.

The following lemma provides an upper bound estimate for the duality gap and the feasibility gap of problem (2).

Lemma 3 *Suppose that $(\bar{x},\bar{y}) \in X \times \mathbb{R}^m$ satisfies the excessive gap condition* (24)*. Then for any $y^* \in Y^*$, we have:*

$$-\|y^*\|\|A\bar{x}-b\| \leq \phi(\bar{x}) - d(\bar{y}) \leq \beta_1(D_1+D_2) - \frac{1}{2\beta_2}\|A\bar{x}-b\|^2 \leq \beta_1(D_1+D_2), \tag{25}$$

and

$$\|A\bar{x}-b\| \leq \beta_2\left[\|y^*\| + \sqrt{\|y^*\|^2 + \frac{2\beta_1}{\beta_2}(D_1+D_2)}\right]. \tag{26}$$

Proof Suppose that $\bar{x}$ and $\bar{y}$ satisfy condition (24). For a given $y^* \in Y^*$, one has:

$$\begin{aligned} d(\bar{y}) \leq d(y^*) &= \min_{x \in X} \left\{ \phi(x) + (Ax-b)^T y^* \right\} \leq \phi(\bar{x}) + (A\bar{x}-b)^T y^* \\ &\leq \phi(\bar{x}) + \|A\bar{x}-b\| \|y^*\|, \end{aligned}$$

which implies the first inequality of (25). By using Lemma 1 and (19) we have:

$$\phi(\bar{x}) - d(\bar{y}) \overset{(17)+(23)}{\leq} f(\bar{x};\beta_2) - d(\bar{y};\beta_1) + \beta_1(D_1+D_2) - \frac{1}{2\beta_2}\|A\bar{x}-b\|^2.$$

Now, by substituting the condition (24) into this inequality, we obtain the second inequality of (25). Let $\eta := \|Ax-b\|$. It follows from (25) that $\eta^2 - 2\beta_2\|y^*\|\eta - 2\beta_1\beta_2(D_1+D_2) \leq 0$. The estimate (26) follows from this inequality after few simple calculations. □

3 New decomposition algorithm

In this section, we derive an iterative decomposition algorithm for solving (2) based on the excessive gap technique. This method is called a *decomposition algorithm with primal update*. The aim is to generate a point $(\bar{x},\bar{y}) \in X \times \mathbb{R}^m$ at each iteration such that this point maintains the excessive gap condition (24) while the algorithm drives the parameters β_1 and β_2 to zero.

3.1 Proximal mappings

As assumed earlier, the function ϕ_i is convex but not necessarily differentiable. Therefore, we can not use the gradient information of these functions. We consider the following mappings ($i = 1,2$):

$$P_i(\hat{x};\beta_2) := \underset{x_i \in X_i}{\operatorname{argmin}} \left\{ \phi_i(x_i) + y^*(\hat{x};\beta_2)^T A_i(x_i - \hat{x}_i) + \frac{L_i^{\psi}(\beta_2)}{2}\|x_i - \hat{x}_i\|^2 \right\}, \tag{27}$$

where $y^*(\hat{x};\beta_2) := \frac{1}{\beta_2}(A\hat{x}-b)$. Since $L_i^{\psi}(\beta_2)$ defined in Lemma 2 is positive, $P_i(\cdot;\beta_2)$ is well-defined. This mapping is called *proximal operator* [7]. Let $P(\cdot;\beta_2) = (P_1(\cdot;\beta_2), P_2(\cdot;\beta_2))$.

First, we state that the excessive gap condition (24) is well-defined by showing that there exists a point $(\bar{x},\bar{y})$ that satisfies (24). This point will be used as a starting point in Algorithm 1 described below.

Lemma 4 *Suppose that $x^c = (x_1^c; x_2^c)$ is the prox-center of X. For a given $\beta_2 > 0$, let us define:*

$$\bar{y} := \frac{1}{\beta_2}(Ax^c - b) \;\text{ and }\; \bar{x} := P(x^c;\beta_2). \tag{28}$$

If the parameter β_1 is chosen such that:

$$\beta_1\beta_2 \geq 2 \max_{1 \leq i \leq 2} \left\{ \frac{\|A_i\|^2}{\sigma_i} \right\}, \tag{29}$$

then $(\bar{x}, \bar{y})$ satisfies the excessive gap condition (24)*.*

The proof of Lemma 4 can be found in the appendix.

3.2 Primal step

Suppose that $(\bar{x},\bar{y}) \in X \times \mathbb{R}^m$ satisfies the excessive gap condition (24). We generate a new point $(\bar{x}^+,\bar{y}^+) \in X \times \mathbb{R}^m$ and by applying the following update scheme:

$$(\bar{x}^+,\bar{y}^+) := \mathscr{A}_m^p(\bar{x},\bar{y};\beta_1,\beta_2^+,\tau) \Longleftrightarrow \left\{ \begin{array}{l} \hat{x} := (1-\tau)\bar{x} + \tau x^*(\bar{y};\beta_1), \\ \bar{y}^+ := (1-\tau)\bar{y} + \tau y^*(\hat{x};\beta_2^+), \\ \bar{x}^+ := P(\hat{x};\beta_2^+), \end{array} \right. \tag{30}$$

$$\beta_1^+ := (1-\tau)\beta_1 \text{ and } \beta_2^+ = (1-\tau)\beta_2, \tag{31}$$

where $P(\cdot;\beta_2^+) = (P_1(\cdot;\beta_2^+), P_2(\cdot;\beta_2^+))$ and $\tau \in (0,1)$ will be chosen appropriately.

Remark 2 In the scheme (30), the points $x^*(\bar{y};\beta_1) = (x_1^*(\bar{y};\beta_1), x_2^*(\bar{y};\beta_1))$, $\hat{x} = (\hat{x}_1,\hat{x}_2)$ and $\bar{x}^+ = (\bar{x}_1^+,\bar{x}_2^+)$ can be computed *in parallel*. To compute $x^*(\bar{y};\beta_1)$ and $\bar{x}^+$ we need to solve the corresponding convex programs in $\mathbb{R}^{n_1}$ and $\mathbb{R}^{n_2}$, respectively.

The following theorem shows that the update rule (30) maintains the excessive gap condition (24).

Theorem 1 *Suppose that $(\bar{x},\bar{y}) \in X \times \mathbb{R}^m$ satisfies* (24) *with respect to two values $\beta_1 > 0$ and $\beta_2 > 0$. Then $(\bar{x}^+,\bar{y}^+)$ generated by scheme* (30)-(31) *is in $X \times \mathbb{R}^m$ and maintains the excessive gap condition* (24) *with respect to two smoothness parameter values β_1^+ and β_2^+ provided that:*

$$\beta_1\beta_2 \geq \frac{2\tau^2}{(1-\tau)^2} \max_{1\leq i\leq 2}\left\{\frac{\|A_i\|^2}{\sigma_i}\right\}. \tag{32}$$

Proof The last line of (30) shows that $\bar{x}^+ \in X$. Let us denote by $\hat{y} = y^*(\hat{x};\beta_2^+)$. Then, by using the definition of $d(\cdot;\beta_1)$, the second line of (30) and $\beta_1^+ = (1-\tau)\beta_1$, we have:

$$\begin{aligned} d(\bar{y}^+;\beta_1^+) &= \min_{x\in X}\left\{\phi(x) + (Ax-b)^T\bar{y}^+ + \beta_1^+[p_1(x_1)+p_2(x_2)]\right\} \\ &\overset{\text{line 2 (30)}}{=} \min_{x\in X}\left\{\phi(x) + (1-\tau)(Ax-b)^T\bar{y} + \tau(Ax-b)^T\hat{y}\right. \\ &\quad \left. + (1-\tau)\beta_1[p_1(x_1)+p_2(x_2)]\right\} \\ &= \min_{x\in X}\left\{(1-\tau)\left[\phi(x) + (Ax-b)^T\bar{y} + \beta_1[p_1(x_1)+p_2(x_2)]\right]\right. \\ &\quad \left. + \tau\left[\phi(x) + (Ax-b)^T\hat{y}\right]\right\}. \end{aligned} \tag{33}$$

Now, we estimate the first term in the last line of (33). Since $\beta_2^+ = (1-\tau)\beta_2$, one has:

$$\psi(\bar{x};\beta_2) = \frac{1}{2\beta_2}\|A\bar{x}-b\|^2 = (1-\tau)\frac{1}{2\beta_2^+}\|A\bar{x}-b\|^2 = (1-\tau)\psi(\bar{x};\beta_2^+). \tag{34}$$

Moreover, if we denote by $x^1 = x^*(\bar{y};\beta_1)$ then, by the strong convexity of p_1 and p_2, (34) and $f(\bar{x};\beta_2) \le d(\bar{y};\beta_1)$, we have:

$$\begin{aligned}
T_1 &:= \phi(x) + (Ax-b)^T\bar{y} + \beta_1[p_1(x_1)+p_2(x_2)] \\
&\ge \min_{x\in X}\left\{\phi(x) + (Ax-b)^T\bar{y} + \beta_1[p_1(x_1)+p_2(x_2)]\right\} + \frac{1}{2}\beta_1\left[\sigma_1\|x_1-x_1^1\|^2 + \sigma_2\|x_2-x_2^1\|^2\right] \\
&= d(\bar{y};\beta_1) + \frac{1}{2}\beta_1\left[\sigma_1\|x_1-x_1^1\|^2 + \sigma_2\|x_2-x_2^1\|^2\right] \\
&\overset{(24)}{\ge} f(\bar{x};\beta_2) + \frac{1}{2}\beta_1\left[\sigma_1\|x_1-x_1^1\|^2 + \sigma_2\|x_2-x_2^1\|^2\right] \\
&\overset{\text{def. } f(\cdot;\beta_2)}{=} \phi(\bar{x}) + \psi(\bar{x};\beta_2) + \frac{1}{2}\beta_1\left[\sigma_1\|x_1-x_1^1\|^2 + \sigma_2\|x_2-x_2^1\|^2\right] \\
&\overset{(34)}{=} \phi(\bar{x}) + \psi(\bar{x};\beta_2^+) + \frac{1}{2}\beta_1\left[\sigma_1\|x_1-x_1^1\|^2 + \sigma_2\|x_2-x_2^1\|^2\right] - \tau\psi(\bar{z};\beta_2^+) \\
&\overset{(22)}{=} \phi(\bar{x}) + \psi(\hat{x};\beta_2^+) + \nabla\psi(\hat{x};\beta_2^+)^T(\bar{x}-\hat{x}) + \frac{1}{2}\beta_1\left[\sigma_1\|x_1-x_1^1\|^2 + \sigma_2\|x_2-x_2^1\|^2\right] \\
&+ \frac{1}{2\beta_2^+}\|A(\bar{x}-\hat{x})\|^2 - \tau\psi(\bar{x};\beta_2^+).
\end{aligned} \tag{35}$$

For the second term in the last line of (33), we use the fact that $\hat{y} = \frac{1}{\beta_2^+}(A\hat{x}-b)$ and $\nabla_y\psi(\hat{x};\beta_2) = A^T\hat{y}$ to obtain:

$$\begin{aligned}
T_2 &:= \phi(x) + (Ax-b)^T\hat{y} \\
&= \phi(x) + \hat{y}^TA(x-\hat{x}) + (A\hat{x}-b)^T\hat{y} \\
&\overset{\text{def. } \hat{y}+(20)}{=} \phi(x) + \nabla\psi(\hat{x};\beta_2^+)^T(x-\hat{x}) + \frac{1}{\beta_2^+}\|A\hat{x}-b\|^2 \\
&\overset{\text{def. } \hat{\psi}}{=} \phi(x) + \psi(\hat{x};\beta_2^+) + \nabla\psi(\hat{x};\beta_2^+)^T(x-\hat{x}) + \psi(\hat{x};\beta_2^+).
\end{aligned} \tag{36}$$

Substituting (35) and (36) into (33) and noting that $(1-\tau)(\bar{x}-\hat{x}) + \tau(x-\hat{x}) = \tau(x-x^1)$ due to the first line of (30), we obtain:

$$\begin{aligned}
d(\bar{y}^+;\beta_1^+) &= \min_{x\in X}\{(1-\tau)T_1 + \tau T_2\} \\
&\overset{(35)+(36)}{\ge} \min_{x\in X}\Big\{(1-\tau)\Big[\phi(\bar{x}) + \psi(\hat{x};\beta_2^+) + \nabla\psi(\hat{x};\beta_2^+)^T(\bar{x}-\hat{x}) \\
&+ \frac{1}{2}\beta_1\left[\sigma_1\|x_1-x_1^1\|^2 + \sigma_2\|x_2-x_2^1\|^2\right]\Big] \\
&+ \tau\left[\phi(x) + \psi(\hat{x};\beta_2^+) + \nabla\psi(\hat{x};\beta_2^+)^T(x-\hat{x})\right]\Big\} \\
&- \tau(1-\tau)\psi(\bar{x};\beta_2^+) + \frac{(1-\tau)}{2\beta_2^+}\|A(\bar{x}-\hat{x})\|^2 + \tau\psi(\hat{x};\beta_2^+) \\
&= \min_{x\in X}\Big\{(1-\tau)\phi(\bar{x}) + \tau\phi(x) + \psi(\hat{x};\beta_2^+) + \nabla\psi(\hat{x};\beta_2^+)^T\left[(1-\tau)(\bar{x}-\hat{x}) + \tau(x-\hat{x})\right] \\
&+ \frac{1}{2}(1-\tau)\beta_1\left[\sigma_1\|x_1-x_1^1\|^2 + \sigma_2\|x_2-x_2^1\|^2\right]\Big\} + \mathbf{T}_3 \\
&\overset{\phi-\text{convex}}{\ge} \min_{x\in X}\Big\{\phi((1-\tau)\bar{x}+\tau x) + \psi(\hat{x};\beta_2^+) + \tau\nabla\psi(\hat{x};\beta_2^+)^T(x-x^1) \\
&+ \frac{1}{2}(1-\tau)\beta_1\left[\sigma_1\|x_1-x_1^1\|^2 + \sigma_2\|x_2-x_2^1\|^2\right]\Big\} + \mathbf{T}_3,
\end{aligned} \tag{37}$$

where $\mathbf{T}_3 := \frac{(1-\tau)}{2\beta_2^+}\|A(\bar{x}-\hat{x})\|^2 + \tau\psi(\hat{x};\beta_2^+) - \tau(1-\tau)\psi(\bar{x};\beta_2^+)$. Next, we note that the condition (32) is equivalent to:

$$(1-\tau)\beta_1\sigma_i \geq \frac{2\tau^2}{(1-\tau)\beta_2}\|A_i\|^2 \geq L_i^{\psi}(\beta_2^+)\tau^2,\ i=1,2. \tag{38}$$

Moreover, if we denote by $u := \bar{x} + \tau(x-\bar{x})$ then:

$$u - \hat{x} = \bar{x} + \tau(x-\bar{x}) - \hat{x} = \bar{x} + \tau(x-\bar{x}) - (1-\tau)\bar{x} - \tau x^1 = \tau(x-x^1). \tag{39}$$

Now, by using Lemma 2, the condition (38) and (39), the estimation (37) becomes:

$$\begin{aligned}
d(\bar{y}^+;\beta_1^+) - \mathbf{T}_3 &\overset{(39)}{\geq} \min_{u:=\bar{x}+\tau(x-\bar{x})\in\bar{x}+\tau(X-\bar{x})} \Big\{\phi(u) + \psi(\hat{x};\beta_2^+) + \nabla\psi(\hat{x};\beta_2)^T(u-\hat{x}) \\
&+ \frac{\beta_1(1-\tau)\sigma_1}{2\tau^2}\|u_1-\hat{x}_1\|^2 + \frac{\beta_1(1-\tau)\sigma_2}{2\tau^2}\|u_2-\hat{x}_2\|^2\Big\} \\
&\overset{\bar{x}+\tau(X-\bar{x})\subseteq X}{\geq} \min_{u\in X}\Big\{\psi(\hat{x};\beta_2^+) + \phi(u) + \nabla\psi(\hat{x};\beta_2^+)^T(u-\hat{x}) \\
&+ \frac{\beta_1(1-\tau)\sigma_1}{2\tau^2}\|u_1-\hat{x}_1\|^2 + \frac{\beta_1(1-\tau)\sigma_2}{2\tau^2}\|u_2-\hat{x}_2\|^2\Big\} \\
&\overset{(38)}{\geq} \min_{u\in X}\Big\{\phi(u) + \psi(\hat{x};\beta_2^+) + \nabla\psi(\hat{x};\beta_2^+)^T(u-\hat{x}) \\
&+ \frac{L_1^{\psi}(\beta_2^+)}{2}\|u_1-\hat{x}_1\|^2 + \frac{L_2^{\psi}(\beta_2^+)}{2}\|u_2-\hat{x}_2\|^2\Big\} \\
&\overset{\text{line 3 (30)}}{=} \phi(\bar{x}^+) + \psi(\hat{x};\beta_2^+) + \nabla\psi(\hat{x};\beta_2^+)^T(\bar{x}^+-\hat{x}) \\
&+ \frac{L_1^{\psi}(\beta_2^+)}{2}\|\bar{x}_1^+-\hat{x}_1\|^2 + \frac{L_2^{\psi}(\beta_2^+)}{2}\|\bar{x}_2^+-\hat{x}_2\|^2 \\
&\overset{(21)}{\geq} \phi(\bar{x}^+) + \psi(\bar{x}^+;\beta_2^+) = f(\bar{x}^+;\beta_2^+).
\end{aligned} \tag{40}$$

To complete the proof, we show that $\mathbf{T}_3 \geq 0$. Indeed, let us define $\hat{u} := A\hat{x} - b$ and $\bar{u} := A\bar{x} - b$, then $\hat{u} - \bar{u} = A(\hat{x}-\bar{x})$. We have:

$$\begin{aligned}
\mathbf{T}_3 &\overset{\text{def. } \psi(\cdot;\beta_2)}{=} \frac{\tau}{2\beta_2^+}\|A\hat{x}-b\|^2 - \frac{\tau(1-\tau)}{2\beta_2^+}\|A\bar{x}-b\|^2 + \frac{(1-\tau)}{2\beta_2^+}\|A(\hat{x}-\bar{x})\|^2 \\
&= \frac{1}{2\beta_2^+}\left[\tau\|\hat{u}\|^2 - \tau(1-\tau)\|\bar{u}\|^2 + (1-\tau)\|\hat{u}-\bar{u}\|^2\right] \\
&= \frac{1}{2\beta_2^+}\left[\tau\|\hat{u}\|^2 - \tau(1-\tau)\|\bar{u}\|^2 + (1-\tau)\|\hat{u}\|^2 + (1-\tau)\|\bar{u}\|^2 - 2(1-\tau)\hat{u}^T\bar{u}\right] \\
&= \frac{1}{2\beta_2^+}\left[\|\hat{u}\|^2 + (1-\tau)^2\|\bar{u}\|^2 - 2(1-\tau)\hat{u}^T\bar{u}\right] \\
&= \frac{1}{2\beta_2^+}\|\hat{u} - (1-\tau)\bar{u}\|^2 \geq 0.
\end{aligned} \tag{41}$$

Substituting (41) into (40) we obtain the inequality $d(\bar{y}^+;\beta_1^+) \geq f(\bar{x}^+;\beta_2^+)$. □

Remark 3 If ϕ_i is convex and differentiable and its gradient is Lipschitz continuous with a Lipschitz constant $L_i^{\phi_i} \geq 0$ for some $i = 1,2$, then instead of using the proximal mapping $P_i(\cdot;\beta_2)$ in (30) we can use the gradient mapping which is defined as:

$$G_i(\hat{x};\beta_2^+) := \underset{x_i \in X_i}{\operatorname{argmin}} \Big\{ \nabla\phi_i(\hat{x}_i)^T(x_i - \hat{x}_i) + y^*(\hat{x};\beta_2)^T A_i(x_i - \hat{x}_i) + \frac{\hat{L}_i^{\psi}(\beta_2^+)}{2}\|x_i - \hat{x}_i\|^2 \Big\}, \quad (42)$$

where $\hat{L}_i^{\psi}(\beta_2^+) := L_{\phi_i} + \frac{2\|A_i\|^2}{\beta_2^+}$. Indeed, let us prove the condition $d(\bar{y}^+;\beta_1^+) \geq f(\hat{x}^+;\beta_2^+)$, where $G(x;\beta_2) := (G_1(x_1;\beta_2), G_2(x_2;\beta_2))$ and $\hat{x}^+ := G(\hat{x};\beta_2^+)$. First, by using the convexity of ϕ_i and the Lipschitz continuity of its gradient, we have:

$$\phi_i(\hat{x}_i) + \nabla\phi_i(\hat{x}_i)^T(u_i - \hat{x}_i) \leq \phi_i(u_i) \leq \phi_i(\hat{x}_i) + \nabla\phi_i(\hat{x}_i)^T(u_i - \hat{x}_i) + \frac{L_{\phi_i}}{2}\|u_i - \hat{x}_i\|^2. \quad (43)$$

Next, by summing up the second inequality from $i = 1$ to 2 and adding to (21) we have:

$$\begin{aligned} \phi(u) + \psi(u;\beta_2^+) \; &\leq \phi(\hat{x}) + \psi(\hat{x};\beta_2^+) + \left[\nabla\phi(\hat{x}) + \nabla\psi(\hat{x};\beta_2^+)\right]^T(u - \hat{x}) \\ &+ \frac{\hat{L}_1^{\psi}(\beta_2^+)}{2}\|u_1 - \hat{x}_1\|^2 + \frac{\hat{L}_2^{\psi}(\beta_2^+)}{2}\|u_2 - \hat{x}_2\|^2. \end{aligned} \quad (44)$$

Finally, from the second inequality of (40) we have:

$$\begin{aligned} d(\bar{y}^+;\beta_1^+) - \mathbf{T}_3 \; &\overset{(38)}{\geq} \min_{u \in X} \Big\{ \phi(u) + \psi(\hat{x};\beta_2^+) + \nabla\psi(\hat{x};\beta_2^+)^T(u - \hat{x}) \\ &+ \frac{(1-\tau)\beta_1\sigma_1}{2\tau^2}\|u_1 - \hat{x}_1\|^2 + \frac{(1-\tau)\beta_1\sigma_2}{2\tau^2}\|u_2 - \hat{x}_2\|^2 \Big\} \\ &\overset{\phi-\text{convex}+(44)}{\geq} \min_{u \in X} \Big\{ \phi(\hat{x}) + \nabla\phi(\hat{x})^T(u - \hat{x}) + \psi(\hat{x};\beta_2^+) + \nabla\psi(\hat{x};\beta_2^+)^T(u - \hat{x}) \\ &+ \frac{\hat{L}_1^{\psi}(\beta_2^+)}{2}\|u_1 - \hat{x}_1\|^2 + \frac{\hat{L}_2^{\psi}(\beta_2^+)}{2}\|u_2 - \hat{x}_2\|^2 \Big\} \\ &\overset{(42)}{=} \phi(\hat{x}) + \psi(\hat{x};\beta_2^+) + \left[\nabla\phi(\hat{x}) + \nabla\psi(\hat{x};\beta_2^+)\right]^T(\hat{x}^+ - \hat{x}) \\ &+ \frac{\hat{L}_1^{\psi}(\beta_2^+)}{2}\|\hat{x}_1^+ - \hat{x}_1\|^2 + \frac{\hat{L}_2^{\psi}(\beta_2^+)}{2}\|\hat{x}_2^+ - \hat{x}_2\|^2 \\ &\overset{(44)}{\geq} \phi(\hat{x}^+) + \psi(\hat{x}^+;\beta_2^+) = f(\hat{x}^+;\beta_2^+). \end{aligned}$$

In this case, the conclusion of Theorem 1 is still valid for the substitution $\hat{x}^+ := G(\hat{x};\beta_2^+)$ provided that:

$$\frac{(1-\tau)}{\tau^2}\beta_1\sigma_i \geq L_{\phi_i} + \frac{2\|A_i\|^2}{(1-\tau)\beta_2}, \; i = 1,2. \quad (45)$$

If X_i is polytopic then problem (42) becomes a convex quadratic programming problem.

Now, let us show how to update the parameter τ such that the condition (32) holds for β_1^+ and β_2^+. From the update rule (31) we have $\beta_1^+\beta_2^+ = (1-\tau)^2\beta_1\beta_2$. Suppose that β_1 and β_2 satisfy the condition (32), i.e.:

$$\beta_1\beta_2 \geq \frac{\tau^2}{(1-\tau)^2}\bar{L}, \text{ where } \bar{L} := 2\max_{1 \leq i \leq 2}\left\{\frac{\|A_i\|^2}{\sigma_i}\right\}.$$

If we substitute β_1 and β_2 by β_1^+ and β_2^+, respectively, in this inequality then we have $\beta_1^+\beta_2^+ \geq \frac{\tau_+^2}{(1-\tau_+)^2}\bar{L}$. However, since $\beta_1^+\beta_2^+ = (1-\tau)^2\beta_1\beta_2$, it implies $\beta_1\beta_2 \geq \frac{\tau_+^2}{(1-\tau)^2(1-\tau_+)^2}\bar{L}$. Therefore, if $\frac{\tau^2}{(1-\tau)^2} \geq \frac{\tau_+^2}{(1-\tau)^2(1-\tau_+)^2}$ then β_1^+ and β_2^+ satisfy (32). This condition leads to $\tau \geq \frac{\tau_+}{1-\tau_+}$. Since $\tau, \tau_+ \in (0,1)$, the last inequality implies $0 < \tau_+ < \frac{1}{2}$ and

$$0 < \tau_+ \leq \frac{\tau}{\tau+1} < 1. \tag{46}$$

Hence, (30)-(31) are well-defined.

Now, we define a rule to update the step size parameter τ.

Lemma 5 *Suppose that τ_0 is arbitrarily chosen in $(0, \frac{1}{2})$. Then the sequence $\{\tau_k\}_{k\geq 0}$ generated by:*

$$\tau_{k+1} := \frac{\tau_k}{\tau_k + 1} \tag{47}$$

satisfies the following equality:

$$\tau_k = \frac{\tau_0}{1+\tau_0 k}, \ \ \forall k \geq 0. \tag{48}$$

Moreover, the sequence $\{\beta_k\}_{k\geq 0}$ generated by $\beta_{k+1} = (1-\tau_k)\beta_k$ for fixed $\beta_0 > 0$ satisfies:

$$\beta_k = \frac{\beta_0}{\tau_0 k + 1}, \ \ \forall k \geq 0. \tag{49}$$

Proof If we denote by $t := \frac{1}{\tau}$ and consider the function $\xi(t) := t + 1$ then the sequence $\{t_k\}_{k\geq 0}$ generated by the rule $t_{k+1} := \xi(t_k) = t_k + 1$ satisfies $t_k = t_0 + k$ for all $k \geq 0$. Hence $\tau_k = \frac{1}{t_k} = \frac{1}{t_0+k} = \frac{\tau_0}{\tau_0 k+1}$ for $k \geq 0$. To prove (49), we observe that $\beta_{k+1} = \beta_0 \prod_{i=0}^{k}(1-\tau_i)$. Hence, by substituting (48) into the last equality and carrying out a simple calculations, we get (49). □

Remark 4 Since $\tau_0 \in (0, 0.5)$, from Lemma 5 we see that with $\tau_0 \to 0.5^-$ (e.g., $\tau_0 = 0.499$) the right-hand side estimate of (49) is minimized.

3.3 The algorithm and its worst case complexity

Before presenting the algorithm, we assume that the prox-center x_i^c of X_i is given a priori for $(i = 1, 2)$. Moreover, the parameter sequence $\{\tau_k\}$ is updated by (47). The algorithm is presented in detail as follows:

ALGORITHM 1 *(Decomposition Algorithm with Primal Update)*

Initialization:

1. Set $\tau_0 := 0.499$. Choose $\beta_1^0 > 0$ and $\beta_2^0 > 0$ as follows:

$$\beta_1^0 = \beta_2^0 := \sqrt{2 \max_{1\leq i\leq 2}\left\{\frac{\|A_i\|^2}{\sigma_i}\right\}}.$$

2. Compute $\bar{x}^0$ and $\bar{y}^0$ from (28) as:

$$\bar{y}^0 := \frac{1}{\beta_2^0}(Ax^c - b) \text{ and } \bar{x}^0 := P(x^c; \beta_2^0),$$

Iteration: `For` $k = 0, 1, \cdots$ `do`

1. If a given stopping criterion is satisfied then terminate.
2. Update the smoothness parameter $\beta_2^{k+1} := (1-\tau_k)\beta_2^k$.
3. Compute $\bar{x}_i^{k+1}$ *in parallel* for $i = 1, 2$ and $\bar{y}^{k+1}$ by the scheme (30):

$$(\bar{x}^{k+1}, \bar{y}^{k+1}) := \mathscr{A}_m^p(\bar{x}^k, \bar{y}^k; \beta_1^k, \beta_2^{k+1}, \tau_k).$$

4. Update the smoothness parameter: $\beta_1^{k+1} := (1-\tau_k)\beta_1^k$.
5. Update the step size parameter τ_k by: $\tau_{k+1} := \frac{\tau_k}{\tau_k+1}$.

`End of For.`

As mentioned in Remark 2, there are two steps of the scheme $\mathscr{A}_m^p$ at Step 3 of Algorithm 1 that can be parallelized. The first step is finding $x^*(\bar{y}^k; \beta_1)$ and the second one is computing $\bar{x}^{k+1}$. In general, both steps require solving two convex programming problems in parallel. The stopping criterion of Algorithm 1 at Step 1 will be discussed in Section 6.

The following theorem provides the worst-case complexity estimate for Algorithm 1.

Theorem 2 *Let $\{(\bar{x}^k, \bar{y}^k)\}$ be a sequence generated by Algorithm 1. Then the following duality gap and feasibility gap hold:*

$$\phi(\bar{x}^k) - d(\bar{y}^k) \leq \frac{\sqrt{\bar{L}}(D_1+D_2)}{0.499k+1}, \tag{50}$$

and

$$\|A\bar{x}^k - b\| \leq \frac{\sqrt{\bar{L}}}{0.499k+1}\left[\|y^*\| + \sqrt{\|y^*\|^2 + 2(D_1+D_2)}\right], \tag{51}$$

where $\bar{L} := 2\max\limits_{1\leq i\leq 2}\left\{\frac{\|A_i\|^2}{\sigma_i}\right\}$ and $y^* \in Y^*$.

Proof By the choice of $\beta_1^0 = \beta_2^0 = \sqrt{\bar{L}}$ and Steps 1 in the initialization phase of Algorithm 1 we see that $\beta_1^k = \beta_2^k$ for all $k \geq 0$. Moreover, since $\tau_0 = 0.499$, by Lemma 5, we have $\beta_1^k = \beta_2^k = \frac{\beta_0}{\tau_0 k+1} = \frac{\sqrt{\bar{L}}}{0.499k+1}$. Now, by applying Lemma 3 with β_1 and β_2 equal to β_1^k and β_2^k respectively, we obtain the estimates (50) and (51). □

Remark 5 The worst case complexity of Algorithm 1 is $O(\frac{1}{\varepsilon})$. However, the constants in the estimations (50) and (51) also depend on the choices of β_1^0 and β_2^0, which satisfy the condition (29). The values of β_1^0 and β_2^0 will affect the accuracy of the duality and feasibility gaps.

In Algorithm 1 we can use a simple update rule $\tau_k = \frac{a}{k+1}$, where $a > 0$ is arbitrarily chosen such that the condition $\tau_{k+1} \leq \frac{\tau_k}{\tau_k+1}$ holds. However, the rule (47) is the tightest one.

4 Switching decomposition algorithm

In this section, we apply the switching strategy to obtain a new variant of the first algorithm proposed in [31, Algorithm 1] for solving problem (2). This scheme alternately switches between the primal and dual step depending on the iteration counter k being even or odd. Apart from its application to Lagrangian dual decomposition, this variant is still different from the one in [31] at two points. First, since we assume that the objective function is not necessarily smooth, instead of using the gradient mapping in the primal scheme, we use the proximal mapping defined by (27) to construct the primal step. In contrast, since the objective function in the dual scheme is Lipschitz continuously differentiable, we can directly use the gradient mapping to compute $\bar{y}^+$ (see (55)). Second, we use the exact update rule for τ instead of the simplified one as in [31].

4.1 The gradient mapping of the smoothed dual function

Since the smoothed dual function $d(\cdot;\beta_1)$ is Lipschitz continuously differentiable on $\mathbb{R}^m$ (see Lemma 1). We define the following mapping:

$$G(\hat{y};\beta_1) := \operatorname*{argmax}_{y\in\mathbb{R}^m}\left\{\nabla d(\hat{y};\beta_1)^T(y-\hat{y}) - \frac{L^d(\beta_1)}{2}\|y-\hat{y}\|^2\right\}, \tag{52}$$

where $L^d(\beta_1) := L_1^d(\beta_1)+L_2^d(\beta_1) = \frac{\|A_1\|^2}{\beta_1\sigma_1} + \frac{\|A_2\|^2}{\beta_1\sigma_2}$ and $\nabla d(\hat{y};\beta_1) = A_1x_1^*(\hat{y};\beta_1)+A_2x_2^*(\hat{y};\beta_1)-b$. This problem can explicitly be solved to get the unique solution:

$$G(\hat{y};\beta_1) = \frac{1}{L^d(\beta_1)}[Ax^*(\hat{y};\beta_1)-b]+\hat{y}. \tag{53}$$

The mapping $G(\cdot;\beta_1)$ is called gradient mapping of the function $d(\cdot;\beta_1)$ (see [29]).

4.2 A decomposition scheme with primal-dual update

First, we adapt the scheme (30)-(31) in the framework of primal and dual variant. Suppose that the pair $(\bar{x},\bar{y}) \in X\times\mathbb{R}^m$ satisfies the excessive gap condition (24). The primal step is computed as follows:

$$(\bar{x}^+,\bar{y}^+) := \mathscr{A}^p(\bar{x},\bar{y};\beta_1,\beta_2,\tau) \iff \begin{cases} \hat{x} := (1-\tau)\bar{x}+\tau x^*(\bar{y};\beta_1), \\ \bar{y}^+ := (1-\tau)\bar{y}+\tau y^*(\hat{x};\beta_2), \\ \bar{x}^+ := P(\hat{x};\beta_2), \end{cases} \tag{54}$$

and then we update $\beta_1^+ := (1-\tau)\beta_1$, where $\tau\in(0,1)$ and $P(\cdot;\beta_2)$ is defined in (27). The difference between schemes $\mathscr{A}_m^p$ and $\mathscr{A}^p$ is that the parameter β_2 is fixed in $\mathscr{A}^p$.

Symmetrically, the dual step is computed as:

$$(\bar{x}^+,\bar{y}^+) := \mathscr{A}^d(\bar{x},\bar{y};\beta_1,\beta_2,\tau) \iff \begin{cases} \hat{y} := (1-\tau)\bar{y}+\tau y^*(\bar{x};\beta_2), \\ \bar{x}^+ := (1-\tau)\bar{x}+\tau x^*(\hat{y};\beta_1), \\ \bar{y}^+ := G(\hat{y};\beta_1), \end{cases} \tag{55}$$

where $\tau \in (0,1)$. The parameter β_1 is kept unchanged, while β_2 is updated by $\beta_2^+ := (1-\tau)\beta_2$.

The following result shows that $(\bar{x}^+, \bar{y}^+)$ generated either by $\mathscr{A}^p$ or by $\mathscr{A}^d$ maintains the excessive gap condition (24).

Lemma 6 *Suppose that $(\bar{x}, \bar{y}) \in X \times \mathbb{R}^m$ satisfy* (24) *with respect to two values β_1 and β_2. Then $(\bar{x}^+, \bar{y}^+)$ generated either by scheme $\mathscr{A}^p$ or by $\mathscr{A}^d$ is in $X \times \mathbb{R}^m$ and maintains the excessive gap condition* (24) *with respect to either two new values β_1^+ and β_2 or β_1 and β_2^+ provided that the following condition holds:*

$$\beta_1\beta_2 \geq \frac{2\tau^2}{1-\tau}\max_{1\leq i\leq 2}\left\{\frac{\|A_i\|^2}{\sigma_i}\right\}. \tag{56}$$

The proof of this lemma is quite similar to [31, Theorem 4.2.] that we omit here.

Remark 6 Given $\beta_1 > 0$, we can choose $\beta_2 > 0$ such that the condition (29) holds. Let $y_c := 0 \in \mathbb{R}^m$, we compute a point $(\bar{x}^0, \bar{y}^0)$ as:

$$\bar{x}^0 := x^*(y^c;\beta_1) \;\text{ and }\; \bar{y}^0 := G(y^c;\beta_1) = \frac{1}{L_d(\beta_1)}(A\bar{x}-c) + y^c. \tag{57}$$

Then, similar to (28), the point $(\bar{x}^0, \bar{y}^0)$ satisfies (24). Therefore, we can use this point as a starting point for Algorithm 2 below.

In Algorithm 2 below we apply either the primal scheme $\mathscr{A}^p$ or the dual scheme $\mathscr{A}^d$ by using the following rule:

Rule A. *If the iteration counter k is even then apply $\mathscr{A}^p$. Otherwise, $\mathscr{A}^d$ is used.*

Now, we provide an update rule to generate a sequence $\{\tau_k\}$ such that the condition (56) holds. Let $\bar{L} := 2\max\limits_{1\leq i\leq 2}\left\{\frac{\|A_i\|^2}{\sigma_i}\right\}$. Suppose that at the iteration k the condition (56) holds, i.e.:

$$\beta_1^k\beta_2^k \geq \frac{\tau_k^2}{1-\tau_k}\bar{L}. \tag{58}$$

Since at the iteration $k+1$, we either update β_1^k or β_2^k. Thus we have $\beta_1^{k+1}\beta_2^{k+1} = (1-\tau_k)\beta_1^k\beta_2^k$. However, as the condition (58) holds, we have $(1-\tau_k)\beta_1^k\beta_2^k \geq \tau_k^2\bar{L}$. Now, we suppose that the condition (56) is satisfied with β_1^{k+1} and β_2^{k+1}, i.e.:

$$\beta_1^{k+1}\beta_2^{k+1} \geq \frac{\tau_{k+1}^2}{1-\tau_{k+1}}\bar{L}. \tag{59}$$

This condition holds if $\tau_k^2\bar{L} \geq \frac{\tau_{k+1}^2}{1-\tau_{k+1}}\bar{L}$, which leads to $\tau_{k+1}^2 + \tau_k^2\tau_{k+1} - \tau_k^2 \leq 0$. Since $\tau_k, \tau_{k+1} \in (0,1)$, we obtain:

$$0 < \tau_{k+1} \leq \frac{\tau_k}{2}\left[\sqrt{\tau_k^2+4} - \tau_k\right] < \tau_k. \tag{60}$$

The tightest rule for updating τ_k is:

$$\tau_{k+1} := \frac{\tau_k}{2}\left[\sqrt{\tau_k^2+4} - \tau_k\right], \tag{61}$$

for all $k \geq 0$ and $\tau_0 \in (0,1)$ given. Associated with $\{\tau_k\}$, we generate two sequences $\{\beta_1^k\}$ and $\{\beta_2^k\}$ as:

$$\beta_1^{k+1} := \begin{cases} (1-\tau_k)\beta_1^k & \text{if } k \text{ is even} \\ \beta_1^k & \text{otherwise,} \end{cases} \text{ and } \beta_2^{k+1} := \begin{cases} \beta_2^k & \text{if } k \text{ is even} \\ (1-\tau_k)\beta_2^k & \text{otherwise,} \end{cases} \tag{62}$$

where $\beta_1^0 = \beta_2^0 = \bar{\beta} > 0$ are fixed.

Lemma 7 *Let $\{\tau_k\}$, $\{\beta_1^k\}$ and $\{\beta_2^k\}$ be three sequences generated by* (61) *and* (62)*, respectively. Then:*

$$\frac{(1-\tau_0)\bar{\beta}}{2\tau_0 k+1} < \beta_1^k < \frac{2\bar{\beta}\sqrt{1-\tau_0}}{\tau_0 k}, \text{ and } \frac{\bar{\beta}\sqrt{1-\tau_0}}{2\tau_0 k+1} < \beta_2^k < \frac{2\bar{\beta}}{\tau_0 k}, \tag{63}$$

for all $k \geq 1$.

The proof of this lemma can be found in the appendix.

Remark 7 We can see that the right-hand side $\eta_k(\tau_0) := \frac{4\bar{\beta}\sqrt{1-\tau_0}}{\tau_0(k+\tau_0)}$ of (63) is decreasing in $(0,1)$ for $k \geq 1$. Therefore, we can choose τ_0 as large as possible to minimize $\eta_k(\cdot)$ in $(0,1)$. For instance, we can choose $\tau_0 := 0.998$ in Algorithm 2.

Note that Lemma 7 shows that $\tau_k \sim O(\frac{1}{k})$. Hence, in Algorithm 2, we can also use a simple updating rule for τ_k as $\tau_k = \frac{a}{k+b}$, where $a \in (\frac{3}{2}, 2)$ and $b \geq \frac{a-1}{2-a} > 0$. This update satisfies (56).

4.3 The algorithm and its worst-case complexity

Suppose that the initial point $(\bar{x}^0, \bar{y}^0)$ is computed by (57). Then, we can choose $\beta_1^0 = \beta_2^0 = \sqrt{2 \max\limits_{1 \leq i \leq 2} \left\{ \frac{\|A_i\|^2}{\sigma_i} \right\}}$ which satisfy (29). The algorithm is now presented in detail as follows:

ALGORITHM 2 *(Decomposition Algorithm with Primal-Dual Update)*

Initialization:

1. Choose $\tau_0 := 0.998$ and set $\beta_1^0 = \beta_2^0 := \sqrt{2\max_{1 \leq i \leq 2} \left\{ \frac{\|A_i\|^2}{\sigma_i} \right\}}$.
2. Compute $\bar{x}^0$ and $\bar{y}^0$ as:

$$\bar{x}^0 := x^*(y^c; \beta_1^0), \text{ and } \bar{y}^0 := \frac{1}{L_d(\beta_1^0)}(A\bar{x}^0 - b) + y^c.$$

Iteration: `For` $k = 0, 1, \cdots$ `do`

1. If a given stopping criterion is satisfied then terminate.
2. If k is even then:
   2a) Compute $(\bar{x}^{k+1}, \bar{y}^{k+1})$ as:

$$(\bar{x}^{k+1}, \bar{y}^{k+1}) := \mathscr{A}^p(\bar{x}^k, \bar{y}^k; \beta_1^k, \beta_2^k, \tau_k).$$

2b) Update the smoothness parameter β_1^k as $\beta_1^{k+1} := (1-\tau_k)\beta_1^k$.

3. Otherwise, i.e. if k is odd then:

3a) Compute $(\bar{x}^{k+1}, \bar{y}^{k+1})$ as:

$$(\bar{x}^{k+1}, \bar{y}^{k+1}) := \mathscr{A}^d(\bar{x}^k, \bar{y}^k; \beta_1^k, \beta_2^k, \tau_k).$$

3b) Update the smoothness parameter β_2^k as $\beta_2^{k+1} := (1-\tau_k)\beta_2^k$.

4. Update the step size parameter τ_k as: $\tau_{k+1} := \frac{\tau_k}{2}\left[\sqrt{\tau_k^2+4} - \tau_k\right]$.

`End of For.`

The main steps of Algorithm 2 are Steps 2a and 2b, which requires us to compute either a primal step or a dual step. In the primal step, we need to solve two convex problem pairs in parallel, while in the dual step, it only requires to solve two convex problems in parallel. The following theorem shows the convergence of this algorithm.

Theorem 3 *Let the sequence $\{(\bar{x}^k, \bar{y}^k)\}_{k\geq 0}$ be generated by Algorithm 2. Then the duality and feasibility gaps satisfy:*

$$\phi(\bar{x}^k) - d(\bar{y}^k) \leq \frac{2\sqrt{\bar{L}}(D_1+D_2)}{0.998k}, \tag{64}$$

and

$$\|A\bar{x}^k - b\| \leq \frac{2\sqrt{\bar{L}}}{0.998k}\left[\|y^*\| + \sqrt{\|y^*\|^2 + 2(D_1+D_2)}\right], \tag{65}$$

where $\bar{L} := 2\max\limits_{1\leq i\leq 2}\left\{\frac{\|A_i\|^2}{\sigma_i}\right\}$ *and* $k \geq 1$.

Proof The conclusion of this theorem follows directly from Lemmas 3 and 5, the condition $\tau_0 = 0.998$, $\beta_1^0 = \beta_2^0 = \sqrt{\bar{L}}$ and the fact that $\beta_1^k \leq \beta_2^k$. □

Remark 8 Note that the worst-case complexity of Algorithm 2 is still $O(\frac{1}{\varepsilon})$. The constants in the complexity estimates (50) and (51) are similar to the one in (64) and (65), respectively. As we discuss in Section 6 below, the rate of decrease of τ_k in Algorithm 2 is smaller than two times of τ_k in Algorithm 1. Consequently, the sequences $\{\beta_1^k\}$ and $\{\beta_2^k\}$ generated by Algorithm 1 approach zero faster than the ones generated by Algorithm 2.

Remark 9 Note that the role of the schemes $\mathscr{A}^p$ and $\mathscr{A}^d$ in Algorithm 2 can be exchanged. Therefore, Algorithm 2 can be modified at three steps to obtain a symmetric variant as follows:

1. At Step 2 of the initialization phase, (28) to compute $\bar{x}^0$ and $\bar{y}^0$ instead of (57).
2. At Steps 2a, $\mathscr{A}^p$ is used if the iteration counter k is odd. Otherwise, we use $\mathscr{A}^d$ at Step 3a.
3. At Steps 2b, β_2^k is updated if k is odd. Otherwise, β_1^k is updated at Step 3b.

5 Application to strongly convex programming problems

If ϕ_i $(i = 1,2)$ in (2) is strongly convex then the convergence rate of the dual scheme (55) can be accelerated up to $O(\frac{1}{k^2})$.

Suppose that ϕ_i is strongly convex with a convexity parameters $\sigma_i > 0$ $(i = 1,2)$. Then the function d defined by (5) is well-defined, concave and differentiable. Moreover, its gradient is given by:

$$\nabla d(y) = A_1 x_1^*(y) + A_2 x_2^*(y) - b, \tag{66}$$

which is Lipschitz continuous with a Lipschitz constant $L^\phi := \frac{\|A_1\|^2}{\sigma_1} + \frac{\|A_2\|^2}{\sigma_2}$. The excessive gap condition (24) in this case becomes:

$$f(\bar{x};\beta_2) \le d(\bar{y}), \tag{67}$$

for given $\bar{x} \in X$, $\bar{y} \in \mathbb{R}^m$ and $\beta_2 > 0$. From Lemma 3 we conclude that if the point $(\bar{x},\bar{y})$ satisfies (67) then, for a given $y^* \in Y^*$, the following estimates hold:

$$-2\beta_2\|y^*\|^2 \le -\|y^*\|\|A\bar{x}-b\| \le \phi(\bar{x}) - d(\bar{y}) \le 0, \tag{68}$$

and

$$\|A\bar{x}-b\| \le 2\beta_2\|y^*\|. \tag{69}$$

We now adapt the dual scheme (55) to this special case. Suppose $(\bar{x},\bar{y}) \in X \times \mathbb{R}^m$ satisfies (67), we generate a new pair $(\bar{x}^+,\bar{y}^+)$ as

$$(\bar{x}^+,\bar{y}^+) := \mathscr{A}_s^d(\bar{x},\bar{y};\beta_2,\tau) \Longleftrightarrow \left\{ \begin{array}{l} \hat{y} := (1-\tau)\bar{y} + \tau y^*(\bar{x};\beta_2), \\ \bar{x}^+ := (1-\tau)\bar{x} + \tau x^*(\hat{y}), \\ \bar{y}^+ = \frac{1}{L^\phi}(Ax^*(\hat{y}) - b) + \hat{y}, \end{array} \right. \tag{70}$$

where $y^*(\bar{x};\beta_2) = \frac{1}{\beta_2}(A\bar{x}-b)$, and $x^*(y) := (x_1^*(y), x_2^*(y))$ is the solution of the minimization problem in (5). The parameter β_2 is updated by $\beta_2^+ := (1-\tau)\beta_2$ and $\tau \in (0,1)$ will appropriately be chosen.

The following lemma shows that $(\bar{x}^+,\bar{y}^+)$ generated by (70) satisfies (67) whose proof can be found in [31].

Lemma 8 *Suppose that the point $(\bar{x},\bar{y}) \in X \times \mathbb{R}^m$ satisfies the excessive gap condition* (67) *with the value β_2. Then the new point $(\bar{x}^+,\bar{y}^+)$ computed by* (70) *is in $X \times \mathbb{R}^m$ and also satisfies* (67) *with a new parameter value β_2^+ provided that*

$$\beta_2 \ge \frac{\tau^2 L_\phi}{1-\tau}. \tag{71}$$

Now, let us derive the rule to update the parameter τ. Suppose that β_2 satisfies (71). Since $\beta_2^+ = (1-\tau)\beta_2$, the condition (71) holds for β_2^+ if $\tau^2 \ge \frac{\tau_+^2}{1-\tau_+}$. Therefore, similar to Algorithm 2, we update the parameter τ by using the rule (47). The conclusion of Lemma 7 still holds for this case.

Before presenting the algorithm, it is necessary to find a starting point $(\bar{x}^0,\bar{y}^0)$ which satisfies (67). Let $y^c = 0 \in \mathbb{R}^m$ and $\beta_2 = L^\phi$. We compute $(\bar{x}^0, \bar{y}^0)$ as

$$\bar{x}^0 := x^*(y^c) \;\text{ and }\; \bar{y}^0 := \frac{1}{L^\phi}(A\bar{x}^0 - b) + y^c. \tag{72}$$

It follows from Lemma 7.4 [31] that $(\bar{x}^0,\bar{y}^0)$ satisfies the excessive gap condition (67).

Finally, the decomposition algorithm for solving the strongly convex programming problem of the form (2) is described in detail as follows:

ALGORITHM 3 *(Decomposition algorithm for strongly convex objective function)*

Initialization:

1. Choose $\tau_0 := 0.5$. Set $\beta_2^0 = \frac{\|A_1\|^2}{\sigma_1} + \frac{\|A_2\|^2}{\sigma_2}$.
2. Compute $\bar{x}^0$ and $\bar{y}^0$ as:

$$\bar{x}^0 := x^*(y^c) \text{ and } \bar{y}^0 := \frac{1}{L^\phi}(A\bar{x}^0 - b) + y^c.$$

Iteration: `For` $k = 0, 1, \cdots$ `do`

1. If a given stopping criterion is satisfied then terminate.
2. Compute $(\bar{x}^{k+1}, \bar{y}^{k+1})$ using scheme (70):

$$(\bar{x}^{k+1}, \bar{y}^{k+1}) := \mathscr{A}_s^d(\bar{x}^k, \bar{y}^k; \beta_2^k, \tau_k).$$

3. Update the smoothness parameter as: $\beta_2^{k+1} := (1-\tau_k)\beta_2^k$.
4. Update the step size parameter τ_k as: $\tau_{k+1} := \frac{\tau_k}{2}\left[\sqrt{\tau_k^2 + 4} - \tau_k\right]$.

`End of For.`

The convergence and the worst-case complexity of Algorithm 3 are stated as in Theorem 4 below.

Theorem 4 *Let $\{(\bar{x}^k, \bar{y}^k)\}_{k\geq 0}$ be a sequence generated by Algorithm 3. Then the following duality and feasibility gaps are satisfied:*

$$-\frac{8L^\phi\|y^*\|^2}{(k+4)^2} \leq \phi(\bar{x}^k) - d(\bar{y}^k) \leq 0, \tag{73}$$

and

$$\|A\bar{x}^k - b\| \leq \frac{8L^\phi\|y^*\|}{(k+4)^2}, \tag{74}$$

where $L^\phi := \frac{\|A_1\|^2}{\sigma_1} + \frac{\|A_2\|^2}{\sigma_2}$.

Proof From the update rule of τ^k, we have $(1-\tau_{k+1}) = \frac{\tau_{k+1}^2}{\tau_k^2}$. Moreover, since $\beta_2^{k+1} = (1-\tau_k)\beta_2^k$, it implies that $\beta_2^{k+1} = \beta_2^0\prod_{i=0}^k(1-\tau_i) = \frac{\beta_2^0(1-\tau_0)}{\tau_0^2}\tau_k^2$. By using the inequalities (80) and $\beta_2^0 = L_\phi$, we have $\beta_2^{k+1} < \frac{4L_\phi(1-\tau_0)}{(\tau_0 k+2)^2}$. With $\tau_0 = 0.5$, one has $\beta_2^k < \frac{8L_\phi}{(k+4)^2}$. By substituting this inequality into (68) and (69), we obtain (73) and (74), respectively. □

Theorem 4 shows that the worst-case complexity of Algorithm 3 is $O(\frac{1}{\sqrt{\varepsilon}})$. Moreover, at each iteration of this algorithm, only two convex problems need to be solved *in parallel*.

6 Discussion on implementation and comparison

6.1 The choice of prox-functions and the Bregman distance

Algorithms 1 and 2 require to build a prox-function for each feasible set X_i for $i = 1, 2$. For a nonempty, closed and bounded convex set X_i, the simplest prox-function is $p_i(x_i) := \frac{\rho_i}{2}\|x_i - \bar{x}_i\|^2$, for a given $\bar{x}_i \in X_i$ and $\rho_i > 0$. This function is strongly convex with the parameter

$\sigma_i = \rho_i$ and the prox-center is $\bar{x}_i$, $(i = 1,2)$. In implementation, it is worth to investigate the structure of the feasible set X_i in order to choose an appropriate prox-function and its scaling factor ρ_i for each feasible subset X_i $(i = 1,2)$.

In (27), we have used the Euclidean distance to construct the proximal terms. It is possible to use a generalized Bregman distance in these problems which is compatible to the prox-function p_i and the feasible subset X_i $(i = 1,2)$. Moreover, a proper choice of the norms in the implementation may lead to a better performance of the algorithms, see [31] for more details.

6.2 Extension to a multi-component separable objective function

The algorithms developed in the previous sections can be directly applied to solve problem (1) in the case $M > 2$. First, we provide the following formulas to compute the parameters of Algorithms 1-3.

1. The constant $\bar{L}$ in Theorems 2 and 3 is replaced by $\bar{L}_M = M \max\limits_{1\le i\le M}\left\{\dfrac{\|A_i\|^2}{\sigma_i}\right\}$.
2. The initial values of β_1^0 and β_2^0 in Algorithms 2 and 3 are $\beta_1^0 = \beta_2^0 = \sqrt{\bar{L}_M}$.
3. The Lipschitz constant $L_i^{\psi}(\beta_2)$ in Lemma 2 is $L_i^{\psi}(\beta_2) = \frac{M\|A_i\|^2}{\beta_2}$ $(i = 1,\ldots,M)$.
4. The Lipschitz constant $L_d(\beta_1)$ in Lemma 1 is $L_d(\beta_1) := \frac{1}{\beta_1}\sum\limits_{i=1}^{M}\dfrac{\|A_i\|^2}{\sigma_i}$.
5. The Lipschitz constant L_ϕ in Algorithm 3 is $L^\phi := \sum\limits_{i=1}^{M}\dfrac{\|A_i\|^2}{\sigma_i}$.

Note that these constants depend linearly on M and the structure of matrix A_i $(i = 1,\ldots,M)$.

Next, we rewrite the smoothed dual function $d(y;\beta_1)$ defined by (11) for the case $M > 2$ as follows:

$$d(y;\beta_1) = \sum_{i=1}^{M} d_i(y;\beta_1),$$

where M function values $d_i(y;\beta_1)$ can be computed in parallel as:

$$d_i(y;\beta_1) = -\frac{1}{M}b_i^T y + \min_{x_i\in X_i}\left\{\phi_i(x_i) + y^T A_i x_i + \beta_1 p_i(x_i)\right\}.$$

Note that the term $-\frac{1}{M}b_i^T y$ is also computed locally for each component subproblem instead of computing separately as in (11). The quantities $\hat{y}$ and $y^+ := G(\hat{y};\beta_1)$ defined in (54) and (55) can respectively be expressed as:

$$\hat{y} := (1-\tau)\bar{y} + (1-\tau)\sum_{i=1}^{M}\frac{1}{\beta_2}(A_i\bar{x}_i - \frac{1}{M}b),$$

$$\text{and } y^+ := \hat{y} + \sum_{i=1}^{M}\left[\frac{1}{L^d(\beta_1)}(A_i x_i^*(\hat{y};\beta_1) - \frac{1}{M}b)\right].$$

These formulas show that each component of $\hat{y}$ and y^+ can be computed by only using the local information and its neighborhood information. Therefore, both algorithms are highly distributed.

Finally, we note that if there exists a component ϕ_i of the objective function ϕ which is Lipschitz continuously differentiable then the gradient projection mapping $G_i(\hat{x};\beta_2)$ defined by (42) corresponding to the primal convex subproblem of this component can be used instead of the proximity mapping $P_i(\hat{x};\beta_2)$ defined by (27). This modification can reduce the computational cost of the algorithms. Note that the sequence $\{\tau_k\}_{k\geq 0}$ generated by the rule (47) still maintains the condition (45) in Remark 3.

6.3 Stopping criterion

In practice, we do not often encounter a problem which reaches the worst-case complexity bound. Therefore, it is necessary to provide a stopping criterion for the implementation of Algorithms 1, 2 and 3 to terminate earlier than using the worst-case bound. In principle, we can use the KKT condition to terminate the algorithms. However, evaluating the global KKT tolerance in a distributed manner is impractical.

From Theorems 2 and 3 we see that the upper bound of the duality and feasibility gaps do not only depend on the iteration counter k but also on the constants $\bar{L}$, D_i and $y^* \in Y^*$. The constant $\bar{L}$ can be explicitly computed based on matrix A and the choice of the prox-functions. We now discuss on the evaluations of D_i and y^* in the case X_i is unbounded. Let sequence $\{(\bar{x}^k,\bar{y}^k)\}$ be generated by Algorithm 1 (or Algorithm 2). Suppose that $\{(\bar{x}^k,\bar{y}^k)\}$ converges to $(x^*,y^*) \in X^* \times Y^*$. Thus, for k sufficiently large, the sequence $\{(\bar{x}^k,\bar{y}^k)\}$ is contained in a neighborhood of $X^* \times Y^*$. Given $\omega > 0$, let us define

$$\hat{D}_i^k := \max_{0\leq j\leq k} p_i(\bar{x}_i^j) + \omega \text{ and } \hat{y}^k := \max_{0\leq j\leq k} \|\bar{y}^j\| + \omega. \tag{75}$$

We can use these constants to construct a stopping criterion in Algorithms 1 and 2. More precisely, for a given tolerance $\varepsilon > 0$, we compute

$$e_d := \beta_1^k(\hat{D}_1^k + \hat{D}_2^k), \text{ and } e_p := \beta_2^k\left[\hat{y}^k + \sqrt{(\hat{y}^k)^2 + 2(\hat{D}_1^k + \hat{D}_2^k)}\right], \tag{76}$$

at each iteration. We terminate Algorithm 1 if $e_d \leq \varepsilon$ and $e_p \leq \varepsilon$. A similar strategy can also be applied to Algorithms 2 and 3.

6.4 Comparison.

Firstly, we compare Algorithms 1 and 2. From Lemma 3 and the proof of Theorems 2 and 3 we see that the rate of convergence of both algorithms is as same as of β_1^k and β_2^k. At each iteration, Algorithm 1 updates simultaneously β_1^k and β_2^k by using the same value of τ_k, while Algorithm 2 updates only one parameter. Therefore, to update both parameters β_1^k and β_2^k, Algorithm 2 needs two iterations. We analyze the update rule of τ_k in Algorithms 1 and 2 to compare the rate of convergence of both algorithms.

Let us define

$$\xi_1(\tau) := \frac{\tau}{\tau+1} \text{ and } \xi_2(\tau) := \frac{\tau}{2}\left[\sqrt{\tau^2+4} - \tau\right].$$

The function ξ_2 can be rewritten as $\xi_2(\tau) = \frac{\tau}{\sqrt{(\tau/2)^2+1}+\tau/2}$. Therefore, we can easily show that:

$$\xi_1(\tau) < \xi_2(\tau) < 2\xi_1(\tau).$$

If we denote by $\{\tau_k^{\mathrm{A}_1}\}_{k\geq 0}$ and $\{\tau_k^{\mathrm{A}_2}\}_{k\geq 0}$ the two sequences generated by Algorithms 1 and 2, respectively then we have $\tau_k^{\mathrm{A}_1} < \tau_k^{\mathrm{A}_2} < 2\tau_k^{\mathrm{A}_1}$ for all k provided that $2\tau_0^{\mathrm{A}_1} \geq \tau_0^{\mathrm{A}_2}$. Since Algorithm 1 updates β_1^k and β_2^k simultaneously while Algorithm 2 updates each of them at each iteration. If we choose $\tau_0^{\mathrm{A}_1} = 0.499$ and $\tau_0^{\mathrm{A}_2} = 0.998$ in Algorithms 1 and 2, respectively, then, by directly computing the value of $\tau_k^{\mathrm{A}_1}$ and $\tau_k^{\mathrm{A}_2}$, we can see that $2\tau_k^{\mathrm{A}_1} > 2\tau_k^{\mathrm{A}_2}$ for all $k \geq 1$. Consequently, the sequences $\{\beta_1^k\}$ and $\{\beta_2^k\}$ in Algorithm 1 converge to zero faster than in Algorithm 2. In other words, Algorithm 1 is faster than Algorithm 2.

Now, we compare Algorithm 1, Algorithm 2 and Algorithm 3.2. in [27] (see also [38]). Note that the smoothness parameter β_1 which is also denoted by c is fixed in Algorithm 3.2 of [27]. Moreover, this parameter is proportional to the given desired accuracy ε, which is often very small. Thus, the Lipschitz constant $L^d(\beta_1)$ is very large. Consequently, Algorithm 3.2. of [27] makes a slow progress at the very early iterations. In Algorithms 1 and 2, the parameters β_1 and β_2 are dynamically updated starting from given values. Besides, the cost per iteration of Algorithm 3.2 [27] is more expensive than Algorithms 1 and 2 since it requires to solve two convex problem pairs in parallel and two dual steps.

7 Numerical Tests

In this section, we verify the performance of the proposed algorithms by applying them to solve the following separable convex optimization problem:

$$\begin{cases} \min\limits_{x=(x_1,\ldots,x_M)} & \Big\{\phi(x) := \sum\limits_{i=1}^{M} \phi_i(x_i)\Big\}, \\ \text{s.t.} & \sum\limits_{i=1}^{M} x_i \leq (=) b, \\ & l_i \leq x_i \leq u_i,\ i = 0,\ldots,M, \end{cases} \tag{77}$$

where $\phi_i : \mathrm{R}^{n_x} \to \mathrm{R}$ is convex, b, l_i and $u_i \in \mathbb{R}^{n_x}$ are given for $i = 1,\ldots,M$. The problem (77) arises in many applications including resource allocation problems [19] and DSL dynamic spectrum management problems [38]. In the case of inequality coupling constraints, we can bring the problem (77) in to the form of (1) by adding a slack variable x_{M+1} as a new component.

7.1 Implementation details

We implement Algorithms 1 and 2 proposed in the previous sections to solve (77). The implementation is carried out in C++ running on a 16 cores workstation Intel®Xeron 2.7GHz and 12 GB of RAM. To solve general convex programming subproblems, we implement a primal-dual predictor-corrector interior point method. All the algorithms are parallelized by using `OpenMP`.

The prox-functions $d_i(x_i) := \frac{\rho}{2}\|x_i - x_i^c\|^2$ are used, where x_i^c is the center of the box $X_i := [l_i, u_i]$ and $\rho := 1$ for all $i = 1,\ldots,M$. We terminate Algorithms 1 and 2 if $\texttt{rpfgap} := \|Ax^k - b\|_2/\|b\|_2 \leq \varepsilon_{\mathrm{p}}$ and either $\texttt{rdfgap} := \max\Big\{0, \beta_1^k \sum_{i=1}^{M} D_{X_i} - \frac{1}{2\beta_2}\|Ax^k - b\|^2\Big\} \leq \varepsilon_{\mathrm{d}}(|\phi(x^k)| + 1)$ or the value of the objective function does not significantly change in 3 successive iterations, i.e. $|\phi(\bar{x}^k) - \phi(\bar{x}^{k-j})|/\max\{1.0, |\phi(\bar{x}^k)|\} \leq \varepsilon_\phi$ for $j = 1,2,3$, where $\varepsilon_{\mathrm{p}} = 10^{-2}$,

$\varepsilon_{\rm d} = 10^{-1}$ and $\varepsilon_\phi = 10^{-5}$ are given tolerances. Note that the quantity `rdfgap` is computed in the worst-case complexity, see Lemma 3.

To compare the performance of the algorithms, we also implement the proximal-center-based decomposition algorithm proposed in [27, Algorithm 3.2.] and an exact variant of the proximal-based decomposition in [7, Algorithm I] for solving (77) which we name `PCBD` and `EPBD`, respectively. The prox-function of the dual problem is chosen as $d_Y(y) := \frac{\rho}{2}\|y\|^2$ with $\rho := 1.0$ and the smoothness parameter c of `PCBD` is set to $c := \frac{\varepsilon_{\rm p}}{\sum_{i=1}^M D_{X_i}}$, where D_{X_i} is defined by (14). We terminate `PCBD` if the relative feasibility gap `rpfgap` $\leq \varepsilon_p$ and either the objective value reaches the one reported by Algorithm 1 or the maximum number of iterations `maxiter` $= 10,000$ is reached.

7.2 Numerical results and comparison

We test the above algorithms for three examples. The two first examples are resource allocation problems and the last one is a DSL dynamic spectrum management problem. The first example was considered in [20], while the problem formulation and the data of the third example are obtained from [38].

7.2.1. Resource allocation problems. Let us consider a resource allocation problem in the form of (77) where the coupling constraint $\sum_{i=1}^M x_i = b$ is tackled.

(a) *Nonsmooth convex optimization problems.* In the first numerical example, we choose $n_x = 1, M = 5$, the objective function $\phi_i(x_i) := i|x_i - i|$ which is nonsmooth and $b = 10$ as in [20]. The lower bound l_i is set to $l_i = -5$ and the upper bound u_i is $u_i = 7$ for $i = 1,\ldots,M$. With these choices, the optimal solution of this problem is $x^* = (-4,2,3,4,5)$.

We use four different algorithms which consist of Algorithm 1, Algorithm 2, `PCBD` in [27] and `PCBD` in [7, Algorithm I] to solved problem (77). The approximate solutions reported by these algorithms after 100 iterations are $x^k = (-3.978,2,3,4,5)$, $(-3.875,1.983, 2.990,3.996,5)$, $(-4.055,2,3,4,5)$ and $(-4.423,2,3,4,5)$, respectively. The corresponding objective values are $\phi(x^k) = 4.978$, 4.954, 5.055 and 5.423, respectively.

The convergence behaviour of four algorithms is shown in Figure 1, where the relative error of the objective function $\mathrm{re}_\phi := |\phi(x^k) - \phi^*|/|\phi^*|$ is plotted on the left and the relative error of the solution $\mathrm{re}_x := \|x^k - x^*\|/\|x^*\|$ is on the right. As we can see from these figures

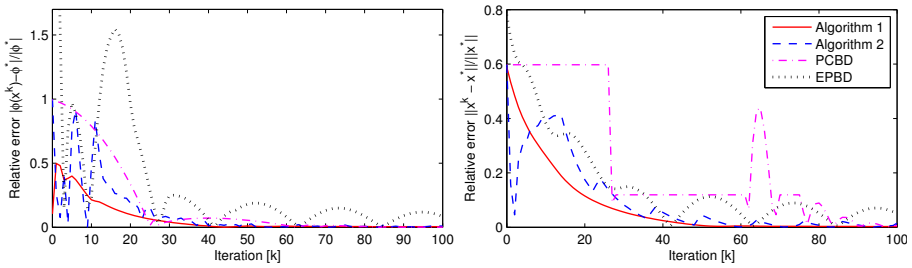

Fig. 1 The relative error of the approximations to the optimal value (left) and to the optimal solution (right).

that the relative errors in Algorithm 2, `PCBD` and `EPBD` oscillate with respect to the iteration counter while they are decreasing monotonously in Algorithm 1. The relative errors in Algorithms 1 and 2 are approaching zero earlier than the ones in `PCBD` and `EPBD`. Note that in this example a nonmonotone variant of the `PCBD` algorithm [27,38] is used.

(b) *Nonlinear resource allocation problems.* In order to compare the efficiency of Algorithm 1, Algorithm 2 and PCBD, we build two performance profiles of these algorithms in terms of total iterations and total computational time.

In this case, the objective function ϕ_i is chosen as $\phi_i(x_i) = a_i^T x_i - w_i \ln(1 + b_i^T x_i)$, where the linear cost vector a_i, vector b_i and the weighting vector w_i are generated randomly in the intervals $[0,5]$, $[0,10]$ and $[0,5]$, respectively. The lower bound and the upper bound are set to $l_i = (0,\ldots,0)^T$ and $u_i = (1,\ldots,1)^T$, respectively. Note that the objective function ϕ_i is linear if $w_i = 0$ and strictly convex if $w_i > 0$.

We carry out three algorithms for solving a collection of 50 random test problems with the size varying from $M = 10$ to $M = 5,000$ components, $m = 5$ to 300 coupling constraints and $n = 50$ to $500,000$ variables. The performance profiles are plotted in Figure 2 which include the total number of iterations (left) and total computational time (right). The nu-

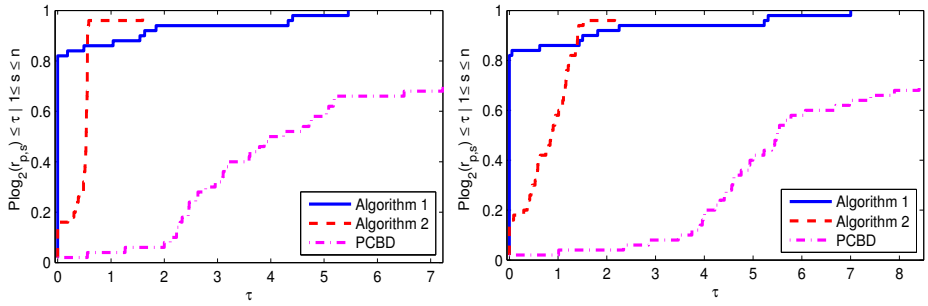

Fig. 2 Performance profile of three algorithms in $\log_2$ scale: Left-Number of iterations, Right-CPU time.

merical test on this collection of problems shows that Algorithm 1 solves all the problems and Algorithm 2 solve $48/50$ problems, i.e. 96% of the collection. PCBD only solves $31/50$ problems, i.e. 62% of the collection. However, Algorithms 1 is the most efficient. It solves up to more than 81% problems with the best performance. PCBD is rather slow and exceeds the maximum number of iterations in many of the test problems (19 problems). Moreover, it is rather sensitive to the smoothness parameter.

7.2.2. DSL dynamic spectrum management problem. In this example, we apply the proposed algorithms to solve a separable convex programming problem arising in DSL dynamic spectrum management. This problem is a convex relaxation of the original DSL dynamic spectrum management formulation considered in [38].

Since the formulation given in [38] has an inequality coupling constraint $\sum_{i=1}^{M} x_i \le b$, by adding a new slack variable x_{M+1} such that $\sum_{i=1}^{M+1} x_i = b$ and $0 \le x_{M+1} \le b$, we can transform this problem into (1). The objective function of the resulting problem becomes:

$$\phi_i(x_i) := \begin{cases} a_i^T x_i - \sum_{j=1}^{n_i} c_i^j \ln\left(\sum_{k=1}^{n_i} h_i^{jk} x_i^k + g_i^k\right) & \text{if } i = 1,\ldots,M, \\ 0 & \text{if } i = M+1. \end{cases} \tag{78}$$

Here, $a_i \in \mathbb{R}^{n_i}$, c_i, $g_i \in \mathbb{R}^{n_i}_+$ and $H_i := (h_i^{jk}) \in \mathbb{R}^{n_i \times n_i}_+$, $(i = 1,\ldots,M)$. The function ϕ_i is convex (but not strongly convex) for all $i = 1,\ldots,M+1$. As described in [38] that the variable x_i is referred to as transmit power spectral density, $n_i = N$ for all $i = 1,\ldots,M$ is the number of users, M is the number of frequency tones which is usually large and ϕ_i is a convex approximation of a desired BER function[1], the coding gain and noise margin. A detail model and parameter descriptions of this problem can be found in [38].

[1] **B**it **E**rror **R**ate function

We test three algorithms for the case of $M = 224$ tones and $N = 7$ users. The other parameters are selected as in [38]. Algorithm 1 requires 922 iterations, Algorithm 2 needs 1314 iterations, while PCBD reaches the maximum number of iterations $k_{\max} = 3000$. The relative feasibility gaps $\|Ax^k - b\|/\|b\|$ reported by the three algorithms are 9.955×10^{-4}, 9.998×10^{-4} and 2.431×10^{-2}, respectively. The obtained approximate solutions of three algorithms and the optimal solution are plotted in Figure 3 which represent the transmit power with respect to the frequency tones. The relative errors of the approximation x^k to the optimal solution x^*, $\texttt{err}_k := \|x^k - x^*\|/\|x^*\|$, are 0.00853, 0.00528 and 0.03264, respectively. The corresponding objective values are 13264.68530, 13259.67633 and 13405.79722, respectively, while the optimal value is 13267.11919.

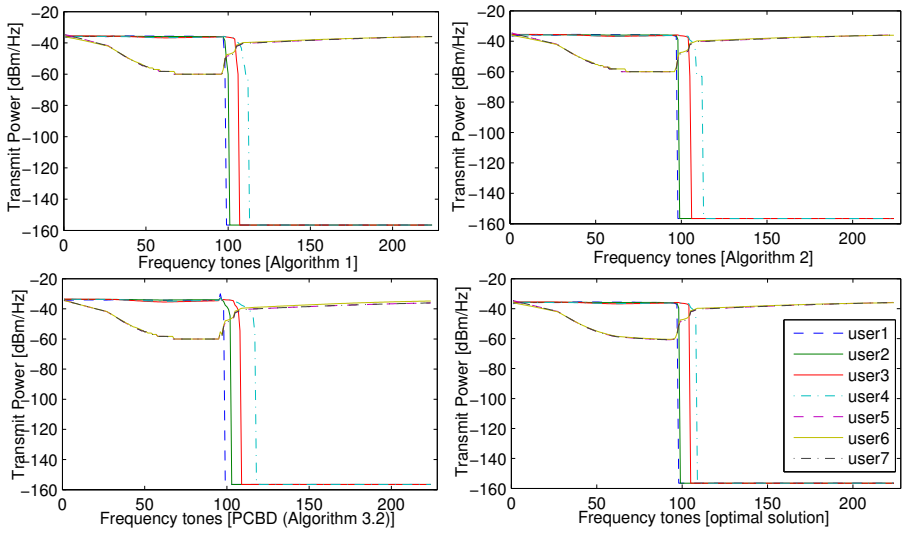

Fig. 3 The approximate solutions of the DSL-dynamic spectrum management problem (77) reported by three algorithms and the optimal solution.

Figure 3 shows that the solutions reported by three algorithms are consistently close to the optimal one. As claimed in [38], PCBD works much better than subgradient methods. However, we can see from this application that Algorithms 1 and 2 require fewer iterations than PCBD to reach a relatively similar approximate solution.

8 Conclusions

In this paper, two new algorithms for large scale separable convex optimization have been proposed. Their convergence has been proved and complexity bound has been given. The main advantage of these algorithms is their ability to dynamically update the smoothness parameters. This allows the algorithms to control the step-size of the search direction at each iteration. Consequently, they generate a larger step at the first iterations instead of remaining fixed for all iterations as in the algorithm proposed in [27]. The convergence behavior and the performance of these algorithms have been illustrated through numerical examples. Although the global convergence rate is still sub-linear, the computational results are remarkable, especially when the number of variables as well as the number of nodes increase.

From a theoretical point of view, the algorithms possess a good performance behavior, due to their numerical robustness and reliability. Currently, the numerical results are still preliminary, however we believe that the theory presented in this paper is useful and may provide guidance for practitioners. Moreover, the steps of the algorithms are rather simple so they can easily be implemented in practice. Future research directions include the dual update scheme and extensions of the algorithms to inexact variants as well as applications.

Acknowledgments. The authors would like to thank Dr. Ion Necoara and Dr. Michel Baes for useful comments on the text and for pointing out some interesting references. Furthermore, the authors are grateful to Dr. Paschalis Tsiaflakis for providing the reality data in the second numerical example. Research supported by Research Council KUL: CoE EF/05/006 Optimization in Engineering(OPTEC), IOF-SCORES4CHEM, GOA/10/009 (MaNet), GOA /10/11, several PhD/postdoc and fellow grants; Flemish Government: FWO: PhD / postdoc grants, projects G.0452.04, G.0499.04, G.0211.05, G.0226.06, G.0321.06, G.0302.07, G.0320.08, G.0558.08, G.0557.08, G.0588.09, G.0377.09, G.0712.11, research communities (ICCoS, ANMMM, MLDM); IWT: PhD Grants, Belgian Federal Science Policy Office: IUAP P6/04; EU: ERNSI; FP7-HDMPC, FP7-EMBOCON, ERC-HIGHWIND, Contract Research: AMINAL. Other: Helmholtz-viCERP, COMET-ACCM.

A. The proofs of Technical Lemmas

This appendix provides the proofs of two technical lemmas stated in the previous sections.

A.1. The proof of Lemma 4. The proof of this lemma is very similar to Lemma 3 in [31].

Proof Let $\hat{y} := y^*(\hat{x};\beta_2) := \frac{1}{\beta_2}(A\hat{x}-b)$. Then it follows from (21) that:

$$\begin{aligned}
\psi(x;\beta_2) &\overset{(21)}{\leq} \psi(\hat{x};\beta_2)+\nabla_1\psi(\hat{x};\beta_2)^T(x_1-\hat{x}_1)+\nabla_2\psi(\hat{x};\beta_2)^T(x_2-\hat{x}_2)\\
&\quad+\frac{L_1^{\psi}(\beta_2)}{2}\|x_1-\hat{x}_1\|^2+\frac{L_2^{\psi}(\beta_2)}{2}\|x_2-\hat{x}_2\|^2\\
&\overset{\text{def. }\psi(\cdot;\beta_2)}{=} \frac{1}{2\beta_2}\|A\hat{x}-b\|^2+\hat{y}^TA_1(x_1-\hat{x}_1)+\hat{y}^TA_2(x_2-\hat{x}_2)+\frac{L_1^{\psi}(\beta_2)}{2}\|x_1-\hat{x}_1\|^2+\frac{L_2^{\psi}(\beta_2)}{2}\|x_2-\hat{x}_2\|^2.\\
&= \hat{y}^T(Ax-b)-\frac{1}{2\beta_2}\|A\hat{x}-b\|^2+\frac{L_1^{\psi}(\beta_2)}{2}\|x_1-\hat{x}_1\|^2+\frac{L_2^{\psi}(\beta_2)}{2}\|x_2-\hat{x}_2\|^2.
\end{aligned} \tag{79}$$

By using the expression $f(x;\beta_2)=\phi(x)+\psi(x;\beta_2)$, the definition of $\bar{x}$, the condition (29) and (79) we have:

$$\begin{aligned}
f(\bar{x};\beta_2) &\overset{(79)}{\leq} \phi(\bar{x})+\bar{y}^TA_1(\bar{x}_1-x_1^c)+\bar{y}^TA_2(\bar{x}_2-x_2^c)\\
&\quad+\frac{L_1^{\psi}(\beta_2)}{2}\|\bar{x}_1-x_1^c\|^2+\frac{L_1^{\psi}(\beta_2)}{2}\|\bar{x}_2-x_2^c\|^2+\frac{1}{2\beta_2}\|Ax^c-b\|^2\\
&\overset{(28)}{=} \min_{x\in X}\Big\{\phi(x)+\frac{1}{\beta_2}\|Ax^c-b\|^2+\bar{y}^TA_1(x_1-x_1^c)+\bar{y}^TA_2(x_2-x_2^c)\\
&\quad+\frac{L_1^{\psi}(\beta_2)}{2}\|x_1-x_1^c\|^2+\frac{L_2^{\psi}(\beta_2)}{2}\|x_2-x_2^c\|^2\Big\}-\frac{1}{2\beta_2}\|Ax^c-b\|^2\\
&= \min_{x\in X}\left\{\phi(x)+\bar{y}^T(Ax-b)+\frac{L_1^{\psi}(\beta_2)}{2}\|x_1-x_1^c\|^2+\frac{L_1^{\psi}(\beta_2)}{2}\|x_2-x_2^c\|^2\right\}-\frac{1}{2\beta_2}\|Ax^c-b\|^2\\
&\overset{(29)}{\leq} \min_{x\in X}\left\{\phi(x)+\bar{y}^T(Ax-b)+\beta_1[p_1(x_1)+p_2(x_2)]\right\}-\frac{1}{2\beta_2}\|Ax^c-b\|^2\\
&= d(\bar{y};\beta_1)-\frac{1}{2\beta_2}\|Ax^c-b\|^2 \leq d(\bar{y};\beta_1),
\end{aligned}$$

which is indeed the condition (24). □

A.2. The proof of Lemma 7.

Proof Let us define $\xi(t) := \frac{2}{\sqrt{1+4/t^2}+1}$. It is easy to show that ξ is increasing in $(0,1)$. Moreover, $\tau_{k+1} = \xi(\tau_k)$ for all $k \geq 0$. Let us introduce $u := 2/t$. Then, we can show that $\frac{2}{u+2} < \xi(\frac{2}{u}) < \frac{2}{u+1}$. By using this inequalities and the increase of ξ in $(0,1)$, we have:

$$\frac{\tau_0}{1+2\tau_0 k} \equiv \frac{2}{u_0+2k} < \tau_k < \frac{2}{u_0+k} \equiv \frac{2\tau_0}{2+\tau_0 k}. \tag{80}$$

Now, by the update rule (62), at each iteration k, we only either update β_1^k or β_2^k. Hence, it implies that:

$$\begin{aligned} \beta_1^k &= (1-\tau_0)(1-\tau_2)\cdots(1-\tau_{2\lfloor k/2 \rfloor})\beta_1^0, \\ \beta_2^k &= (1-\tau_1)(1-\tau_3)\cdots(1-\tau_{2\lfloor k/2 \rfloor - 1})\beta_2^0, \end{aligned} \tag{81}$$

where $\lfloor x \rfloor$ is the largest integer number which is less than or equal to the positive real number x. On the other hand, since $\tau_{i+1} < \tau_i$ for $i \geq 0$, for any $l \geq 0$, it implies:

$$\begin{aligned} &(1-\tau_0)\textstyle\prod_{i=0}^{2l}(1-\tau_i) < [(1-\tau_0)(1-\tau_2)\cdots(1-\tau_{2l})]^2 < \textstyle\prod_{i=0}^{2l+1}(1-\tau_i), \\ \text{and } &\textstyle\prod_{i=0}^{2l-1}(1-\tau_i) < [(1-\tau_1)(1-\tau_3)\cdots(1-\tau_{2l-1})]^2 < (1-\tau_0)^{-1}\textstyle\prod_{i=0}^{2l}(1-\tau_i). \end{aligned} \tag{82}$$

Note that $\prod_{i=0}^{k}(1-\tau_i) = \frac{(1-\tau_0)}{\tau_0^2}\tau_k^2$, it follows from (81) and (82) for $k \geq 1$ that:

$$\frac{(1-\tau_0)\beta_1^0}{\tau_0}\tau_{k+1} < \beta_1^{k+1} < \frac{\beta_1^0\sqrt{1-\tau_0}}{\tau_0}\tau_{k-1}, \text{ and } \frac{\beta_2^0\sqrt{1-\tau_0}}{\tau_0}\tau_{k+1} < \beta_2^{k+1} < \frac{\beta_2^0}{\tau_0}\tau_{k-1}.$$

By combining these inequalities and (80), and noting that $\tau_0 \in (0,1)$, we obtain (63). □

References

1. Alexandre, d'A., Onureena, B., and Laurent, E.G.: First-order methods for sparse covariance selection. SIAM J. Matrix Anal. Appl. **30**(1), 56–66 (2008).
2. Bertsekas, D.P., and Tsitsiklis, J.N.: *Parallel and Distributed Computation: Numerical Methods*. Englewood Cliffs, NJ: Prentice-Hall, (1989).
3. Bertsekas, D.P.: *Constrained Optimization and Lagrange Multiplier Methods*. Athena Scientific, Belmont, Massachusetts (1996).
4. Bertsekas, D.P.: Incremental proximal methods for large-scale convex optimization. Report LIDS - 2847 (2010).
5. Bienstock, D., and Iyengar, G.: Approximating fractional packings and coverings in $O(1/\varepsilon)$ iterations. SIAM J. Comput. **35**(4), 825–854 (2006).
6. Boyd, S., Parikh, N., Chu, E., Peleato, B. and Eckstein, J.: Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. Foundations and Trends in Machine Learning, **3**:1, 1-122 (2011).
7. Chen, G., and Teboulle, M.: A proximal-based decomposition method for convex minimization problems. Math. Program., **64**, 81–101 (1994).
8. Cohen, G.: Optimization by decomposition and coordination: A unified approach. IEEE Trans. Automat. Control, **AC-23**(2), 222–232 (1978).
9. Connejo, A. J., Mínguez, R., Castillo, E. and García-Bertrand, R.: *Decomposition Techniques in Mathematical Programming: Engineering and Science Applications*. Springer-Verlag, (2006).
10. Eckstein, J. and Bertsekas, D.: On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. Math. Program. **55**, 293–318 (1992).
11. Fukushima, M., Haddou, M., Van Hien, N., Strodiot, J.J., Sugimoto, T., and Yamakawa, E.: A parallel descent algorithm for convex programming. Comput. Optim. Appl. **5**(1), 5–37 (1996).
12. Goldfarb, D., and Ma, S.: Fast Multiple Splitting Algorithms for Convex Optimization. SIAM J. on Optim., (submitted) (2010).
13. Hamdi, A.: Decomposition for structured convex programs with smooth multiplier methods. Applied Mathematics and Computation, 169, 218–241 (2005).

14. Hans-Jakob, L., and Jörg, D.: Convex risk measures for portfolio optimization and concepts of flexibility. Math. Program., **104**(2-3), 541–559 (2005).
15. Han, S.P., and Lou, G.: A Parallel Algorithm for a Class of Convex Programs. SIAM J. Control Optim. **26**, 345-355 (1988).
16. Hariharan, L., and Pucci, F.D.: Decentralized resource allocation in dynamic networks of agents. SIAM J. Optim. **19**(2), 911–940 (2008).
17. Holmberg, K.: Experiments with primal-dual decomposition and subgradient methods for the uncapacitated facility location problem. Optimization **49**(5-6), 495–516 (2001).
18. Holmberg, K. and Kiwiel, K.C.: Mean value cross decomposition for nonlinear convex problem. Optim. Methods and Softw. **21**(3), 401–417 (2006).
19. Ibaraki, T. and Katoh, N.: *Resource Allocation Problems: Algorithmic Approaches: Foundations of Computing*. The MIT Press (1988).
20. Johansson, B. and Johansson, M.: Distributed non-smooth resource allocation over a network. Proc. IEEE conference on Decision and Control, 1678–1683, (2009).
21. Kojima, M., Megiddo, N. and Mizuno, S. et al: Horizontal and vertical decomposition in interior point methods for linear programs. Technical Report. Information Sciences, Tokyo Institute of Technology (1993).
22. Komodakis, N., Paragios, N., and Tziritas, G.: MRF Energy Minimization & Beyond via Dual Decomposition. IEEE Transactions on Pattern Analysis and Machine Intelligence (in press).
23. Kontogiorgis, S., Leone, R.D., and Meyer, R.: Alternating direction splittings for block angular parallel optimization. J. Optim. Theory Appl., **90**(1), 1–29 (1996).
24. Love, R.F., and Kraemer, S.A.: A dual decomposition method for minimizing transportation costs in multifacility location problems. Transportation Sci. **7**, 297–316 (1973).
25. Mehrotra, S. and Ozevin, M. G.: Decomposition Based Interior Point Methods for Two-Stage Stochastic Convex Quadratic Programs with Recourse. Operation Research, **57**(4), 964–974 (2009).
26. Neveen, G., Jochen, K.: Faster and simpler algorithms for multicommodity flow and other fractional packing problems. SIAM J. Comput. **37**(2), 630–652 (2007).
27. Necoara, I. and Suykens, J.A.K.: Applications of a smoothing technique to decomposition in convex optimization, IEEE Trans. Automatic control, **53**(11), 2674–2679 (2008).
28. Nesterov, Y.: *A method for unconstrained convex minimization problem with the rate of convergence* $O(1/k^2)$. Doklady AN SSSR 269, 543–547 (1983); translated as Soviet Math. Dokl.
29. Nesterov, Y.: *Introductory Lectures on Convex Optimization*. Kluwer, Boston (2004).
30. Nesterov, Y.: Smooth minimization of nonsmooth functions. Math. Program., 103(1):127–152, (2005).
31. Nesterov, Y.: Excessive gap technique in nonsmooth convex minimization, SIAM J. Optimization, **16**(1), 235–249, (2005).
32. Purkayastha, P., and Baras, J.S.: An optimal distributed routing algorithm using dual decomposition techniques. Commun. Inf. Syst. **8**(3), 277–302 (2008).
33. Ruszczyński, A.: On convergence of an augmented Lagrangian decomposition method for sparse convex optimization. Mathematics of Operations Research, **20**, 634–656 (1995).
34. Samar, S., Boyd, S., and Gorinevsky,D.: Distributed Estimation via Dual Decomposition. Proceedings European Control Conference (ECC), 1511–1516, Kos, Greece, (2007).
35. Spingarn, J.E.: Applications of the method of partial inverses to convex programming: Decomposition. Math. Program. Ser. A, **32**, 199–223 (1985).
36. Tran Dinh, Q., Necoara, I., Savorgnan, C. and Diehl, M.: An Inexact Perturbed Path-Following Method for Lagrangian Decomposition in Large-Scale Separable Convex Optimization. Tech. Report, 1–37, (2011), url: http://arxiv.org/abs/1109.3323.
37. Tseng, P.: Alternating projection-proximal methods for convex programming and variational inequalities. SIAM J. Optim. **7**(4), 951–965 (1997).
38. Tsiaflakis P., Necoara I., Suykens J.A.K., Moonen M.: Improved Dual Decomposition Based Optimization for DSL Dynamic Spectrum Management. IEEE Transactions on Signal Processing, **58**(4), 2230–2245, (2010).
39. Vania Dos Santos Eleuterio: *Finding Approximate Solutions for Large Scale Linear Programs*. PhD Thesis, No 18188, ETH Zurich, (2009).
40. Venkat, A., Hiskens, I., Rawlings, J., and Wright, S.: Distributed MPC strategies with application to power system automatic generation control. IEEE Trans. Control Syst. Technol. **16**(6), 1192–12-6 (2008).
41. Zhao, G.: A Lagrangian dual method with self-concordant barriers for multistage stochastic convex programming. Math. Progam. **102**, 1–24 (2005).