Regularized Jacobi-type ADMM-Methods for a Class of Separable Convex Optimization Problems in Hilbert Spaces

Eike Börgens* Christian Kanzow*

October 2, 2017

Abstract

We consider a regularized version of a Jacobi-type alternating direction method of multipliers (ADMM) for the solution of a class of separable convex optimization problems in a Hilbert space. The analysis shows that this method is equivalent to the standard proximal-point method applied in a Hilbert space with a transformed scalar product. The method therefore inherits the known convergence results from the proximal-point method and allows suitable modifications to get a strongly convergent variant. Some additional properties are also shown by exploiting the particular structure of the ADMM-type solution method. Applications and numerical results are provided to the sparse optimization problem in finite dimensions and to the domain decomposition method in a Hilbert space setting.

1 Introduction

We consider the separable problem

$$\min \sum_{i=1}^{N} f_i(x_i) \quad \text{s.t.} \quad \sum_{i=1}^{N} A_i x_i = b,\ x_i \in \mathcal{X}_i\ (i = 1, \ldots, N), \tag{1}$$

where $\mathcal{H}_i$ and $\mathcal{K}$ are Hilbert spaces, $f_i : \mathcal{H}_i \to \mathbb{R}$ are lower semi-continuous, convex functions, $\mathcal{X}_i \subset \mathcal{H}_i$ are closed convex sets, and $A_i \in \mathcal{L}(\mathcal{H}_i, \mathcal{K})$, $b \in \mathcal{K}$. We assume that the optimization problem (1) has a nonempty feasible set. Note that all functions f_i are supposed to be convex only, none of them has to be strictly or uniformly convex. Furthermore, no differentiability of f_i is required. Since we have explicit constraints $\mathcal{X}_i$ for

*University of Würzburg, Institute of Mathematics, Campus Hubland Nord, Emil-Fischer-Str. 30, 97074 Würzburg, Germany; {eike.boergens,kanzow}@mathematik.uni-wuerzburg.de.

each mapping f_i, there is no loss of generality to have f_i real-valued for all $i = 1, \ldots, N$. Moreover, we do not assume the operators A_i to be injective or surjective, a condition that is often used in finite dimensions where the matrices A_i are assumed to have full rank.

For the sake of notational simplicity, we use the abbreviations

$$\begin{aligned} \mathcal{H} &:= \mathcal{H}_1 \times \cdots \times \mathcal{H}_N, \qquad \mathcal{X} := \mathcal{X}_1 \times \ldots \times \mathcal{X}_N \subseteq \mathcal{H} \\ x &:= (x_1, \ldots, x_N) \in \mathcal{H}, \qquad f(x) := \sum_{i=1}^{N} f_i(x_i), \qquad Ax := \sum_{i=1}^{N} A_i x_i. \end{aligned}$$

Canonically $\mathcal{H}$ becomes a Hilbert space with the scalar product $\langle x \mid y \rangle := \langle x_1 \mid y_1 \rangle + \cdots + \langle x_N \mid y_N \rangle$, the scalar product in the space $\mathcal{H} \times \mathcal{K}$ is defined analogously. The symbol $\|\cdot\|$ always denotes the norm induced by the corresponding scalar product (in $\mathcal{H}_i, \mathcal{H}, \mathcal{K}$, or $\mathcal{H} \times \mathcal{K}$); the meaning should be clear from the context.

Using this notation, we can rewrite (1) as

$$\min_x \ f(x) \quad \text{s.t.} \quad Ax = b, \ x \in \mathcal{X}. \tag{2}$$

Let

$$\begin{aligned} L(x, \mu) &:= f(x) + \langle \mu \mid Ax - b \rangle, \\ L_A(x, \mu) &:= f(x) + \langle \mu \mid Ax - b \rangle + \frac{\beta}{2} \|Ax - b\|^2 \end{aligned}$$

denote the Lagrangian and the augmented Lagrangian of (2), respectively, where $\beta > 0$ is the penalty parameter. Then a standard optimization technique for solving optimization problems of this kind is the augmented Lagrangian or multiplier-penalty method, in the following abbreviated by ALM. The basic iteration of ALM applied to (2) is given by

$$x^{k+1} := \operatorname*{arg\,min}_{x \in \mathcal{X}} L_A(x, \mu^k), \qquad \mu^{k+1} := \mu^k + \beta(Ax^{k+1} - b),$$

provided that a minimum of the augmented Lagrangian exists and can be computed (hopefully) easily, cf. [3, 21, 26].

Unfortunately, when applied to the separable problem (1), the quadratic term in the augmented Lagrangian destroys the separable structure and, therefore, ALM cannot take advantage of the separability in the computation of the new iterate x^{k+1}. This observation is the main motivation for the alternating direction method of multipliers, ADMM for short. This method computes the block components x_i^{k+1} essentially again by minimizing $L_A(x, \mu^k)$, but with the full-dimensional vector x being replaced either by $(x_1^k, \ldots, x_{i-1}^k, x_i, x_{i+1}^k, \ldots, x_N^k)$ or by $(x_1^{k+1}, \ldots, x_{i-1}^{k+1}, x_i, x_{i+1}^k, \ldots, x_N^k)$, so that the subproblems are minimization problems in x_i alone. More precisely, discarding some constant terms, the former approach leads to a Jacobi-type ADMM method with x_i^{k+1} being computed by

$$x_i^{k+1} := \operatorname*{arg\,min}_{x \in \mathcal{X}_i} \left\{ f_i(x_i) + \langle \mu^k \mid A_i x_i \rangle + \frac{\beta}{2} \Big\| A_i x_i + \sum_{l \neq i} A_l x_l^k - b \Big\|^2 \right\} \tag{3}$$

for all $i = 1, \ldots, N$, whereas the latter approach yields the Gauss-Seidel-type ADMM method

$$x_i^{k+1} := \operatorname*{arg\,min}_{x \in \mathcal{X}_i} \Big\{ f_i(x_i) + \langle \mu^k \mid A_i x_i \rangle + \frac{\beta}{2} \Big\| \sum_{l<i} A_l x_l^{k+1} + A_i x_i + \sum_{l>i} A_l x_l^k - b \Big\|^2 \Big\}. \quad (4)$$

Both methods coincide for the case $N = 1$ and reduce to the standard ALM approach, whereas the classical ADMM method corresponds to the Gauss-Seidel-type iteration (4) with $N = 2$ blocks.

Note that the two schemes (3) and (4) have different properties. The former is implementable completely in parallel, whereas the latter is not, but uses the newer information and is therefore often faster convergent in terms of the number of outer iterations. In any case, both methods have the major advantage that they can fully exploit the separable structure of (1) and often yield small-dimensional subproblems that are easy to solve (sometimes even analytically). Unfortunately, however, without any further assumptions, these subproblems might not have solutions, and even if they have, none of the two schemes necessarily converges. In fact, while there is a satisfactory global convergence theory for the Gauss-Seidel-type ADMM-scheme for the special case $N = 2$, see [4, 12], the recent paper [6] shows that convergence cannot be expected, in general, for $N > 2$. For the Jacobi-type scheme the situation is even worse since [17] provides a counterexample for the case of $N = 2$ blocks.

It therefore comes with no surprise that there exist a couple of modifications of the two basic iterations (3) and (4) in the literature. We refer the interested reader to [18, 19] and references therein for some suitable modifications of the Gauss-Seidel-type scheme, and to [7, 15, 17, 20, 29, 30] for the Jacobi-type method. Since our aim is to present a modification of the Jacobi-type iteration, we concentrate our discussion on this class of methods, some more details will be given in Section 3 after an explicit statement of our algorithm.

To the best of our knowledge, the existing literature (see citations above) that investigates the convergence properties of suitable modifications of the Jacobi-type scheme (3) is exclusively written in the finite-dimensional setting. All methods that do not regularize the x_i-subproblems require the suboperators A_i to be injective (full rank assumption) in order to be well-defined and to get convergence of the iterates $\{x^k\}$, whereas this assumption is not necessarily needed in an approach that regularizes the subproblems, cf. [1, 7].

The method we present here is not completely new. In fact, during the finalization of this paper we became aware of the very recent publication [7] that considers a parallel multi-block ADMM scheme which is essentially the same as the algorithm considered here. Nevertheless, there are some differences which we think are remarkable. First, we present our method and the corresponding theory in a Hilbert space setting, whereas [7] considers finite-dimensional problems. Second, we reduce our convergence theory to a standard proximal-point approach, as opposed to [7] where the authors provide an independent (self-contained) convergence theory. Third, the lower bounds for certain parameters used here and in [7] seem to be better (smaller) in our theory, which in turn

leads to a superior numerical behavior. Finally, we address certain questions (like weak and strong convergence and an application in Hilbert spaces) which do not occur in the finite-dimensional theory.

The paper is organized as follows: Some basic definitions and preliminary results are stated in Section 2. Our regularized Jacobi-type ADMM-method is presented in Section 3 together with a more detailed discussion regarding some of the above-cited related algorithms. The corresponding global convergence analysis is given in Section 4. The main idea is to show that, after a linear transformation, it is equivalent to a sequence generated by a proximal-point method in a suitable Hilbert space. This transformation is possible for the Jacobi-type iteration and is not directly applicable to the corresponding Gauss-Seidel-version of our approach. Motivated by the proximal-point interpretation of our algorithm, which only gives weak convergence of the iterates unless additional assumptions hold, we present a strongly convergent Halpern-type modification of the Jacobi-type ADMM method in Section 5. Section 6 describes the domain decomposition technique for a partial differential equation as (an infinite-dimensional) application of our method. Some numerical results are presented in Section 7, both for the previously considered domain decomposition technique and for the (finite-dimensional) sparse optimization problem. We conclude with some final remarks in Section 8.

2 Definitions and Preliminaries

We first recall some notions and definitions from set-valued and variational analysis. For more details, we refer the reader to the excellent monograph [2].

The *distance function* in a Hilbert space $\mathcal{H}$ of a point $w \in \mathcal{H}$ to a nonempty, closed, and convex set $C \subset \mathcal{H}$ is given by $\operatorname{dist}(C, w) := \inf_{v \in C} \|w - v\|_{\mathcal{H}}$. Given a set-valued operator $T : \mathcal{H} \to 2^{\mathcal{H}}$, the *domain* of T is defined by $\operatorname{dom} T := \{x \in \mathcal{H} \mid T(x) \neq \emptyset\}$, the *graph* of T is given by $\operatorname{graph}(T) := \big\{(x, u) \in \mathcal{H} \times \mathcal{H} \mid u \in T(x)\big\}$, and the *inverse* of T is defined through its graph $\operatorname{graph}(T^{-1}) := \big\{(u, x) \in \mathcal{H} \times \mathcal{H} \mid u \in T(x)\big\}$, so that $u \in T(x)$ if and only if $x \in T^{-1}(u)$. The set-valued operator T is called *monotone* if

$$\langle u - v \mid x - y \rangle \geq 0 \qquad \forall (x, u), (y, v) \in \operatorname{graph} T,$$

and *maximally monotone* if it is monotone and its graph is not properly contained in the graph of a monotone operator. A set-valued operator T is called strongly monotone when there is a constant $c > 0$ such that

$$\langle u - v \mid x - y \rangle \geq c\|x - y\|^2 \qquad \forall (x, u), (y, v) \in \operatorname{graph} T.$$

The *convex subdifferential* ∂f of a convex function $f : \mathcal{H} \to (-\infty, +\infty]$ is defined by

$$\partial f(\bar{x}) := \big\{s \in \mathcal{H} \mid f(x) - f(\bar{x}) \geq \langle s \mid x - \bar{x} \rangle \ \forall x \in \mathcal{H}\big\},$$

and is known to be maximally monotone.

An operator $T : \mathcal{H} \to \mathcal{H}$ is said bo be *firmly non-expansive* if

$$\|Tx - Ty\|^2 \leq \langle Tx - Ty \mid x - y \rangle \quad \forall x, y \in \mathcal{H},$$

and *non-expansive* if it is Lipschitz continuous with constant 1, i.e. $\|Tx - Ty\| \leq \|x - y\|$ for all $x, y \in \mathcal{H}$. The Cauchy-Schwarz inequality shows that every firmly non-expansive operator is also non-expansive.

We call a function $f : \mathcal{H} \to (-\infty, +\infty]$ *lower semi-continuous* (lsc) in $x \in \mathcal{H}$ if for every net $x_a \to x$ it holds that $\liminf f(x_a) \geq f(x)$, and *weakly sequentially lsc* in $x \in \mathcal{H}$ if for every weakly convergent sequence $x^k \rightharpoonup x$ we have $\liminf_{k\to\infty} f(x_k) \geq f(x)$. The function f is called (weakly sequentially) lsc (in $\mathcal{H}$) if it is (weakly sequentially) lsc in all points $x \in \mathcal{H}$. Recall that a convex function f is weakly sequentially lsc if and only if it is lsc, cf. [2, Thm. 9.1]. The *normal cone* of a vector $x \in \mathcal{X}$ is defined by

$$N_{\mathcal{X}}(x) := \begin{cases} \{s \in \mathcal{H} \mid \langle s \mid y - x \rangle \leq 0 \ \forall y \in \mathcal{X}\} & \text{if } x \in \mathcal{X} \\ \emptyset & \text{if } x \notin \mathcal{X} \end{cases}.$$

This notation allows us to define the standard notion of a KKT point.

Definition 2.1. A pair $(x^*, \mu^*) \in \mathcal{X} \times \mathcal{K}$ is called a *KKT point* of (1) if it satisfies the following *KKT conditions*: $0 \in \partial f(x) + A^*\mu + N_{\mathcal{X}}(x)$ and $0 = b - Ax$, where $A^* : \mathcal{K} \to \mathcal{H}$ denotes the Hilbert space adjoint of A.

Note that a KKT point has to be feasible with respect to the abstract constraints $\mathcal{X}$, whereas they exploit the existence of a multiplier for the equality constraints. This setting is useful for our ADMM-type method where only the linear constraints are penalized, whereas the abstract constraints remain unchanged.

Our aim is to compute a KKT point of the optimization problem (1). In many cases, this is equivalent to finding a solution of the minimization problem itself. More precisely, the KKT conditions are always sufficient optimality conditions, whereas the necessary part usually requires some constraint qualifications; for example, $b \in \operatorname{sri} A(\mathcal{X})$, see [2, Prop. 27.14], where sri denotes the strong relative interior, see [2, Def. 6.9]. In the finite-dimensional case the condition $Z \cap \operatorname{int}\mathcal{X} \neq \emptyset$ would be enough for the KKT conditions to be necessary optimality conditions, where $Z := \{x \mid Ax = b\}$, cf. [27, Cor. 28.2.2] for a more detailed discussion. This constraint qualification holds, in particular, when $\mathcal{X}_i = \mathcal{H}_i$ for all $i = 1, \ldots, N$.

In order to rewrite the KKT conditions in a more compact form, let us further introduce the notation

$$\mathcal{W} := \mathcal{X}_1 \times \ldots \times \mathcal{X}_N \times \mathcal{K}, \quad w := \big(x_1, \ldots, x_N, \mu\big), \quad f(w) := f(x),$$

where the last expression simply means that, depending on the argument, we either view f as a mapping depending on x only, or depending on the full vector $w = (x, \mu)$. Therefore, for the corresponding subdifferentials (with respect to w and x, respectively), depending on the corresponding arguments, we have

$$\partial f(w) = \begin{pmatrix} \partial f(x) \\ \{0\} \end{pmatrix},$$

since f is independent of μ. Finally, let us define $G : \mathcal{H} \times \mathcal{K} \to \mathcal{H} \times \mathcal{K}$,

$$G(w) := \begin{pmatrix} A_1^*\mu \\ A_2^*\mu \\ \vdots \\ A_N^*\mu \\ b - \sum_{i=1}^N A_i x_i \end{pmatrix}. \tag{5}$$

The particular structure of G immediately yields the following result.

Lemma 2.2. *The mapping G as defined in (5) satisfies $\langle G(w) - G(\bar{w}) \mid w - \bar{w} \rangle = 0$ for all $w, \bar{w} \in \mathcal{W}$; in particular, G is a continuous monotone operator.*

The above notation yields the following compact representation of the KKT conditions.

Lemma 2.3. *The vector pair $w^* = (x^*, \mu^*) \in \mathcal{X} \times \mathcal{K}$ is a KKT point of (1) if and only if $w^* \in \mathcal{W}^*$, where $\mathcal{W}^* := \{w \in \mathcal{W} \mid 0 \in \partial f(w) + G(w) + N_{\mathcal{W}}(w)\}$.*

Proof. The proof follows immediately from the previous definitions, taking into account that, due to the Cartesian structure of $\mathcal{W}$, we have

$$N_{\mathcal{W}}(w) = N_{\mathcal{X}_1}(x_1) \times \ldots \times N_{\mathcal{X}_N}(x_N) \times N_{\mathcal{K}}(\mu)$$

and $N_{\mathcal{K}}(\mu) = \{0\}$ since $\mathcal{K}$ is the entire space. □

Let us define the multifunction

$$T(w) := \partial f(w) + G(w) + N_{\mathcal{W}}(w), \tag{6}$$

whose domain is obviously given by the nonempty set $\mathcal{W}$. Then the set $\mathcal{W}^*$ from Lemma 2.3 can be expressed as $\mathcal{W}^* = \{w \in \mathcal{W} \mid 0 \in T(w)\}$. This indicates that the set-valued-mapping T plays a central role in our analysis. Its most important property is formulated in the following result.

Proposition 2.4. *The set-valued function T defined in (6) is maximal monotone.*

Proof. Since G is a continuous monotone function in view of Lemma 2.2 and $\operatorname{dom} G = \mathcal{H} \times \mathcal{K}$, it follows that $B := G + N_{\mathcal{W}}$ is a maximal monotone operator, see, e.g., [2, Cor. 25.5]. Furthermore, since f is a real-valued convex function, it is also known that $A := \partial f(w)$ is maximal monotone. Since $\operatorname{dom}(A) = \operatorname{dom}(\partial f) = \mathcal{H} \times \mathcal{K}$, see [2, Cor. 8.39 combined with Prop. 16.17] and $\operatorname{dom}(B) \neq \emptyset$, it follows again from [2, Cor. 25.5] that $T = \partial f + G + N_{\mathcal{W}} = A + B$ is also maximal monotone. □

Another way to verify the maximal monotonicity of T is through the maximal monotonicity of the convex-concave subdifferential, cf. [1].

3 Regularized Jacobi-type ADMM-Method

The method we consider in this paper is the following regularized Jacobi-type method to a separable version of the augmented Lagrangian approach for the solution of the optimization problem (1).

Algorithm 3.1. *(Regularized Jacobi-type ADMM Method)*

(S.0) Choose a starting point $(x^0, \mu^0) \in \mathcal{X} \times \mathcal{K}$*, parameters* $\beta, \gamma > 0$*, and set* $k := 0$*.*

(S.1) If a suitable termination criterion is satisfied: STOP.

(S.2) For $i = 1, \ldots, N$*, compute*

$$x_i^{k+1} := \operatorname*{arg\,min}_{x_i \in \mathcal{X}_i} \Big\{ f_i(x_i) + \langle \mu^k \mid A_i x_i \rangle + \frac{\beta}{2} \big(\|A_i x_i + \sum_{l \neq i} A_l x_l^k - b\|^2 + \gamma \|x_i - x_i^k\|^2 \big) \Big\}. \quad (7)$$

(S.3) Define

$$\mu^{k+1} := \mu^k + \beta \Big(\sum_{l=1}^{N} A_l x_l^{k+1} - b \Big). \quad (8)$$

(S.4) Set $k \leftarrow k+1$*, and go to (S.1).*

Throughout our convergence analysis, we assume implicitly that Algorithm 3.1 generates an infinite number of iterates. We further note that all subproblems (7) are strongly convex for all i and all iterations k. Hence $x^{k+1} := \big(x_1^{k+1}, \ldots, x_N^{k+1}\big)$ is uniquely defined. Note that this is due to the quadratic regularization term which does not occur in standard ADMM-methods for two or more components. These standard ADMM-methods are also Gaussian-type methods since they use the newer information $x_1^{k+1}, \ldots, x_{i-1}^{k+1}$ in the computation of x_i^{k+1} in (7). For reasons that will become clear during our convergence analysis, we use the above Jacobi-type ADMM-method with its known advantages and disadvantages.

The main computational overhead in Algorithm 3.1 comes from the solution of the optimization subproblems in (S.2). However, in contrast to the ALM method, these subproblems are small-dimensional. Moreover, there are several applications where these subproblems can be solved analytically, in which case each iteration of the algorithm is extremely cheap.

In order to compare our method with some existing ones and to present some suitable modifications, let us denote the outcome of (S.2) also by $\hat{x}_i^k$, i.e.

$$\hat{x}_i^k := \operatorname*{arg\,min}_{x_i \in \mathcal{X}_i} \Big\{ f_i(x_i) + \langle \mu^k \mid A_i x_i \rangle + \frac{\beta}{2} \big(\|A_i x_i + \sum_{l \neq i} A_l x_l^k - b\|^2 + \gamma \|x_i - x_i^k\|^2 \big) \Big\}. \quad (9)$$

Furthermore, when there is no partial regularization, we denote the corresponding result by

$$\tilde{x}_i^k := \operatorname*{arg\,min}_{x_i \in \mathcal{X}_i} \Big\{ f_i(x_i) + \langle \mu^k \mid A_i x_i \rangle + \frac{\beta}{2} \big(\|A_i x_i + \sum_{l \neq i} A_l x_l^k - b\|^2 \big) \Big\}. \quad (10)$$

This allows us to state the following comments, where we mainly concentrate on some modified updates of the iterates x^k, but it should be clear that corresponding updates are then also needed for the multiplier μ^k.

Remark 3.2. In the following we discuss some related algorithms from the existing literature; recall that the convergence proofs of all these methods are done in the finite-dimensional case only.

(a) The modified Jacobi-type ADMM-method suggested in [17] uses the iteration $x^{k+1} := x^k + \alpha_k(\tilde{x}^k - x^k)$ for some step size $\alpha_k > 0$. This step size can either be computed by a formula or is constant and explicitly given, but typically very small. We further note that the paper [17] needs all sub-matrices A_i to be of full column rank.

(b) Motivated by the previous comment, it might also be useful to rewrite Algorithm 3.1 as $x^{k+1} := x^k + \alpha_k(\hat{x}^k - x^k)$ for some step size $0 < c_l \leq \alpha_k \leq c_u < 2$. Obviously, Algorithm 3.1 corresponds to the case $\alpha_k = 1$, i.e. we can allow much larger step sizes than [17]. This does not automatically guarantee faster convergence, especially since we have the additional regularization term in our method, but indicates that there is some hope for a superior numerical behavior. We will get back to this step size later.

(c) Recall that Algorithm 3.1 was already analyzed in [7] for the finite-dimensional case, whereas we deal with the Hilbert space setting and state some further results (e.g., strong convergence). Our results, based on a very simple technique of proof, therefore generalize those from [7]. Furthermore, as noted in (b), our approach also allows under- and overrelaxation of the iterates (x^k, μ^k), whereas the technique in [7] allows to introduce a step size $\tau \in (0, 2)$ only in the dual variable μ^k. The corresponding μ^k-update gets

$$\mu^{k+1} = \mu^k + \tau\beta\Big(\sum_{i=1}^{N} A_i x_i^{k+1} - b\Big).$$

A simplified condition on the proximal constant γ_i for the i-th subproblem given in [7, Lem. 2.2] is

$$\gamma_i I \succ \big(\frac{N}{2-\tau} - 1\big) A_i^T A_i,$$

where $A \succ B$ means that $A - B$ is positive definite. In our approach, it is also possible to choose the proximal constant separately for every subproblem or even choosing a different equivalent norm for the regularization as in [7], but the added value would be minimal compared to the notational inconvenience. Moreover, taking into account that, in the finite-dimensional case, the matrix $N \cdot \mathrm{diag}(A_1^T A_1, \ldots, A_N^T A_N) - A^T A$ is easily seen to be positive semi-definite, it follows that $(N-1) \cdot \mathrm{diag}(A_1^T A_1, \ldots, A_N^T A_N) \succeq A^T A - \mathrm{diag}(A_1^T A_1, \ldots, A_N^T A_N)$, where $B \succeq C$ means that $B - C$ is positive semi-definite. Consequently, for γ large enough we have

$$0 \preceq \gamma I - (N-1) \cdot \mathrm{diag}(A_1^T A_1, \ldots, A_N^T A_N) \preceq \gamma I - (A^T A - \mathrm{diag}(A_1^T A_1, \ldots, A_N^T A_N)),$$

hence the condition from [7, Lem. 2.2] implies our condition on γ that we will introduce later in Section 4. Taking A as the identity matrix, we see that our criterion regarding the choice of γ can indeed be significantly weaker.

(d) Another modification of the Jacobi-type iteration (3) is due to [15, 30] and replaces the update from (10) by

$$\tilde{x}_i^k := \operatorname*{arg\,min}_{x_i \in \mathcal{X}_i} \Big\{ f_i(x_i) + \langle \mu^k \mid A_i x_i \rangle + \frac{\beta}{2} \big(\|A_i x_i + \sum_{l \neq i} A_l x_l^k - b\|^2 + (N-1) \|A_i(x_i - x_i^k)\|^2 \big) \Big\}$$

which can also be re-interpreted as a partial regularization method involving the matrix A_i in the regularization term, cf. [15, Alg. 8.1] or, as an application of the two function ADMM on a modified problem, see [30]. Theorem 4.1 from [20] shows that these two different approaches yield the same algorithm. Consequently, this modification also requires a full rank assumption on each A_i to be well-defined and to get convergence of the iterates x^k.

(e) An algorithm that is also basically parallel was introduced in [29] and uses the scheme

$$\begin{aligned}
\hat{x}_1^k &:= \operatorname*{arg\,min}_{x_1 \in \mathcal{X}_1} \Big\{ f_1(x_1) + \langle \mu^k \mid A_1 x_1 \rangle + \frac{\beta}{2} \big(\|A_1 x_1 + \sum_{l \neq 1} A_l x_l^k - b\|^2 \big) \Big\}, \\
\hat{\mu}^k &:= \mu^k + \beta (A_1 \hat{x}_1^k + \sum_{l=2}^N A_l x_l^k), \\
\hat{x}_i^k &:= \operatorname*{arg\,min}_{x_i \in \mathcal{X}_i} \Big\{ f_i(x_i) + \langle \mu^k \mid A_i x_i \rangle + \frac{\beta}{2} \big(\|A_i x_i + A_1 \hat{x}_1^k + \sum_{\substack{l=2 \\ l \neq i}}^N A_l x_l^k\|^2 + \|x_i - x_i^k\|_{M_i}^2 \big) \Big\}
\end{aligned}$$

for all $i = 2, \ldots, N$, where M_i are some positive definite matrices that satisfy some condition. The analysis carried out in [29], however, is completely different from ours and requires, similar to [7], a choice of certain parameters related to the matrices M_i which seem to be less favorable from a numerical point of view, cf. the results in Section 7.

4 Convergence Analysis

The main idea of our convergence analysis is to interpret Algorithm 3.1, after a simple linear transformation, as a proximal-point method applied to a suitable inclusion problem in an appropriate Hilbert space.

To this end, let us introduce the linear operator $M \in \mathcal{L}(\mathcal{H}) := \mathcal{L}(\mathcal{H}, \mathcal{H})$ by

$$Mx := \Big(\sum_{\substack{l=1 \\ l \neq i}}^N A_i^* A_l x_l \Big)_{i=1}^N = \begin{pmatrix} \sum_{l=2}^N A_1^* A_l x_l \\ \vdots \\ \sum_{\substack{l=1 \\ l \neq i}}^N A_i^* A_l x_l \\ \vdots \\ \sum_{l=1}^{N-1} A_N^* A_l x_l \end{pmatrix}. \tag{11}$$

In finite dimensions, the representation matrix of M is given by

$$M := A^T A - \operatorname{diag}(A_1^T A_1, \ldots, A_N^T A_N).$$

Further define $Q \in \mathcal{L}(\mathcal{H} \times \mathcal{K})$ by

$$Q \begin{pmatrix} x \\ \mu \end{pmatrix} := \begin{pmatrix} \beta^2(\gamma x - Mx) \\ \mu \end{pmatrix}, \tag{12}$$

where β, γ denote the constants from Algorithm 3.1. In finite dimensions, the matrix representation of Q is

$$Q := \begin{pmatrix} \beta^2(\gamma I - M) & 0 \\ 0 & I \end{pmatrix}.$$

The following simple remark plays a crucial role in our subsequent convergence analysis.

Remark 4.1. By definition, the operator M from (11) is self-adjoint. Hence Q from (12) is also self-adjoint. Moreover, for all $\gamma > 0$ sufficiently large (say $\gamma > \|M\|$), Q is also strongly monotone, i.e. there is a constant $c > 0$ such that $\langle Qx \mid x \rangle \geq c\|x\|^2$. This implies that Q is both injective and coercive, hence also surjective. Hence the inverse of $Q \in \mathcal{L}(\mathcal{H} \times \mathcal{K})$ exists and is also a linear, continuous and self-adjoint operator.

Our first result gives a suitable reformulation for the optimality conditions of the subproblems from (7) and (8).

Lemma 4.2. *The vector $w^{k+1} = \big(x^{k+1}, \mu^{k+1}\big)$ computed in* (7) *and* (8) *is characterized by the inclusion*

$$0 \in \beta \partial f(w^{k+1}) + \beta G(w^{k+1}) + Q(w^{k+1} - w^k) + N_{\mathcal{W}}(w^{k+1}). \tag{13}$$

Proof. Using the optimality conditions for the programs (7), it follows that x_i^{k+1} solves these programs if and only if $x_i = x_i^{k+1}$ satisfies the optimality conditions

$$0 \in \partial_{x_i} \Big(f_i(x_i) + \langle \mu^k \mid A_i x_i \rangle + \frac{\beta}{2} \big(\|A_i x_i + \sum_{l \neq i} A_l x_l^k - b\|^2 + \gamma \|x_i - x_i^k\|^2 \big) \Big) + N_{\mathcal{X}_i}(x_i)$$

for all $i = 1, \ldots, N$. This is equivalent to saying that there exist elements $g_i \in \partial f_i(x_i^{k+1})$ such that

$$-\Big(g_i + A_i^* \mu^k + \beta A_i^* (A_i x_i^{k+1} + \sum_{l \neq i} A_l x_l^k - b) + \beta\gamma(x_i^{k+1} - x_i^k) \Big) \in N_{\mathcal{X}_i}(x_i^{k+1})$$

for all $i = 1, \ldots, N$. By definition of the normal cone, this can be rewritten as

$$\Big\langle g_i + A_i^* \mu^k + \beta A_i^* \big(A_i x_i^{k+1} + \sum_{l \neq i} A_l x_l^k - b \big) + \beta\gamma(x_i^{k+1} - x_i^k) \mid x_i - x_i^{k+1} \Big\rangle \geq 0 \tag{14}$$

for all $x_i \in \mathcal{X}_i$ and all $i = 1, \dots, N$. Using $\mu^{k+1} = \mu^k + \beta\big(\sum_{l=1}^N A_l x_l^{k+1} - b\big)$, the last inequality is equivalent to

$$\Big\langle g_i + A_i^* \mu^{k+1} + \beta A_i^*\big(\sum_{l \neq i} A_l(x_l^k - x_l^{k+1})\big) + \beta\gamma(x_i^{k+1} - x_i^k) \mid x_i - x_i^{k+1} \Big\rangle \geq 0$$

for all $x_i \in \mathcal{X}_i$ and all $i = 1, \dots, N$. Exploiting the definition of M in (11), the Cartesian product structure of the set $\mathcal{X}$, and setting $\tilde{g} = (g_1, \dots, g_N)$, this can be rewritten more compactly as

$$\Big\langle \tilde{g} + A^* \mu^{k+1} + \beta M(x^k - x^{k+1}) + \beta\gamma(x^{k+1} - x^k) \mid x - x^{k+1} \Big\rangle \geq 0$$

for all $x \in \mathcal{X}$. Since

$$\Big\langle \frac{1}{\beta}(\mu^{k+1} - \mu^k) + \big(b - \sum_{i=1}^N A_i x_i^{k+1}\big) \mid \mu - \mu^k \Big\rangle = 0 \qquad \forall \mu \in \mathcal{K}$$

in view of (8), the previous two formulas can be rewritten as

$$\Big\langle g + G(w^{k+1}) + \begin{pmatrix} \beta(\gamma I - M) & 0 \\ 0 & \frac{1}{\beta} I \end{pmatrix} (w^{k+1} - w^k) \mid w - w^{k+1} \Big\rangle \geq 0 \quad \forall w \in \mathcal{W},$$

where $g := \big(g_1, \dots, g_N, 0\big)$. Multiplication with β and taking into account the definition of Q from (12) yields

$$\Big\langle \beta g + \beta G(w^{k+1}) + Q(w^{k+1} - w^k) \mid w - w^{k+1} \Big\rangle \geq 0 \quad \forall w \in \mathcal{W}.$$

Using the definition of the normal cone $N_{\mathcal{W}}$, we can express this as

$$0 \in \beta g + \beta G(w^{k+1}) + Q(w^{k+1} - w^k) + N_{\mathcal{W}}(w^{k+1}).$$

Since $g \in \partial f(w^{k+1})$ we see that this is equivalent to

$$0 \in \beta \partial f(w^{k+1}) + \beta G(w^{k+1}) + Q(w^{k+1} - w^k) + N_{\mathcal{W}}(w^{k+1}).$$

This completes the proof. □

In the following, we will use the previous characterization of the stationary points to get an equivalent procedure for the computation of the iterates $w^{k+1} := \big(x^{k+1}, \mu^{k+1}\big)$ from Algorithm 3.1. To this end, we assume throughout that Q is strongly monotone, which is always possible for sufficiently large (and computable) γ, cf. Remark 4.1.

Lemma 4.3. *If Q is invertible and T maximal monotone we have that*

$$\big(I + \beta Q^{-1} T\big)^{-1} = (Q + \beta T)^{-1} Q$$

Proof. The definition of the pre-image immediately yields

$$\begin{aligned} v \in \big(I + \beta Q^{-1}T\big)^{-1} w &\iff w \in \big(I + \beta Q^{-1}T\big) v \\ &\iff Qw \in \big(Q + \beta T\big) v \\ &\iff v \in \big(Q + \beta T\big)^{-1} Qw \end{aligned}$$

for all $w \in \mathcal{H} \times \mathcal{K}$. □

Based on the previous results, we obtain the following alternative procedure for the computation of w^{k+1} from Algorithm 3.1.

Proposition 4.4. *Assume that Q is self adjoint and strongly monotone. Given an iterate $w^k = \big(x^k, \mu^k\big)$, the next iterate $w^{k+1} := \big(x^{k+1}, \mu^{k+1}\big)$ generated by Algorithm 3.1 can equivalently be represented by the (single-valued) formula*

$$w^{k+1} := \big(I + \beta Q^{-1}T\big)^{-1} w^k, \tag{15}$$

where $Q^{-1}T$ and $\big(I+\beta Q^{-1}T\big)^{-1}$ are maximal monotone and $\big(I+\beta Q^{-1}T\big)^{-1}$ is firmly non-expansive in the Hilbert space $\mathcal{H} \times \mathcal{K}$ endowed with the scalar product $\langle x \mid y \rangle_Q := \langle Qx \mid y \rangle$.

Proof. First recall that the iterate w^{k+1} computed by Algorithm 3.1 is uniquely defined. Furthermore, due to convexity, they are fully characterized by the optimality conditions (13) from Lemma 4.2. In order to rewrite these optimality conditions, recall that $N_{\mathcal{W}}(w^{k+1})$ is a cone, so we have $\beta N_{\mathcal{W}}(w^{k+1}) = N_{\mathcal{W}}(w^{k+1})$. The definition of the operator T from (6) therefore allows us to rewrite the inclusion from (13) as

$$\begin{aligned} 0 \in \beta T(w^{k+1}) + Q(w^{k+1} - w^k) &= \big(Q + \beta T\big)(w^{k+1}) - Qw^k \\ \iff\ & Qw^k \in \big(Q + \beta T\big)(w^{k+1}) \\ \iff\ & w^{k+1} \in \big(I + \beta Q^{-1}T\big)^{-1} w^k, \end{aligned}$$

where the last equivalence exploits Lemma 4.3. We claim that the last inclusion is actually an equation, from which we then obtain the assertion. To this end, recall that T is a maximal monotone operator in view of Proposition 2.4. Hence βT is also maximal monotone. It is obvious that Q^{-1} is self-adjoint and strongly monotone, thus by [2, Prop. 20.24] it follows that $\beta Q^{-1}T$ is maximal monotone in the Hilbert space $\mathcal{H} \times \mathcal{K}$ endowed with the scalar product $\langle \cdot \mid \cdot \rangle_Q$, hence its resolvent is single-valued. Finally, notice that the resolvent of a maximal monotone operator is always maximally monotone and firmly non-expansive, cf. [2, Cor. 23.11]. □

Note that Proposition 4.4 uses two different scalar products (and therefore two different induced norms) for our Hilbert space $\mathcal{H} \times \mathcal{K}$. In order to apply the known convergence results of the proximal-point method, it is highly important in our setting that both strong and weak convergence are identical concepts in both settings. Formally, this is stated in the following result.

Lemma 4.5. *Consider the Hilbert space $\mathcal{H} \times \mathcal{K}$ with the usual scalar product $\langle \cdot \mid \cdot \rangle$, and let $\langle \cdot \mid \cdot \rangle_Q$ be defined as in Proposition 4.4 for Q self-adjoint and strongly monotone. Then the following statements hold:*

(a) The corresponding induced norms $\|\cdot\|$ and $\|\cdot\|_Q$ are equivalent.

(b) Weak convergence with respect to $\langle \cdot \mid \cdot \rangle$ is equivalent to weak convergence with respect to $\langle \cdot \mid \cdot \rangle_Q$.

Proof. (a) Since Q is linear, bounded, and strongly monotone, there exist constants $0 < c \leq C$ such that $c\|x\|^2 \leq \langle Qx \mid x \rangle \leq C\|x\|^2$. This immediately yields statement (a).

(b) This statement follows from

$$\begin{aligned} \langle x, y\rangle_Q \to \langle x^\infty, y\rangle_Q \quad \forall y \quad &\iff \quad \langle Qx, y\rangle \to \langle Qx^\infty, y\rangle \quad \forall y \\ &\iff \quad \langle x, Qy\rangle \to \langle x^\infty, Qy\rangle \quad \forall y \\ &\iff \quad \langle x, z\rangle \to \langle x^\infty, z\rangle \quad \forall z, \end{aligned}$$

where the last two equivalences exploit that Q is self-adjoint and invertible, cf. Remark 4.1. □

The previous results allow us to re-interpret Algorithm 3.1 as a generalized proximal-point method and to inherit the convergence properties from the known convergence properties of proximal-point methods.

Theorem 4.6. *Suppose that Q is self-adjoint, strongly monotone, and that there is at least one KKT point for the optimization problem* (1)*. Then the following statements hold:*

(a) The sequence $\{w^k\} = \{(x^k, \mu^k)\}$ generated by Algorithm 3.1 converges weakly to a KKT point $w^\infty = (x^\infty, \mu^\infty)$ of (1)*, i.e. $0 \in T(w^\infty)$, where x^∞ is a solution of the optimization problem* (1)*.*

(b) It holds that $\|w^k - w^{k+1}\|^2 = O(1/k)$, in particular, $\|w^k - w^{k+1}\| \to 0$ for $k \to \infty$.

(c) It holds that $Ax^k - b \to 0$ for $k \to \infty$ and $Ax^\infty = b$ (primal feasibility).

(d) We have $\operatorname{dist}^2(T(w^k), 0) = o(1/k)$ for $k \to \infty$ (rate of convergence to KKT-optimality).

(e) It holds that $f(x^k) \to f^*$, where f^* is the optimal function value of (1)*.*

(f) We have $f_i(x_i^k) \to f_i(x_i^\infty)$, where x_i^∞ is the weak limit point of x_i^k, for all $i = 1, \ldots, N$.

(g) If f_i is strongly convex, then x_i^k converges strongly to x_i^∞.

Proof. (a) By Proposition 4.4, we know that $Q^{-1}T$ is maximal monotone in the Hilbert space $\mathcal{H} \times \mathcal{K}$ endowed with the scalar product $\langle \cdot \mid \cdot \rangle_Q$. Thus the proximal point method converges weakly in this Hilbert space to a zero of $Q^{-1}T$, see [2, Thm. 23.41]. But the zeros of $Q^{-1}T$ are the same as the zeros of the operator T and therefore correspond to a KKT point of the underlying optimization problem, cf. Lemma 2.3. Furthermore, Lemma 4.5 implies that we also have weak convergence with respect to the orignal scalar product $\langle \cdot \mid \cdot \rangle$. Hence statement (a) follows.

(b) This assertion is a consequence of the equivalence of our norms together with the convergence results stated in [9, Thm. 3.1].

(c) By definition of μ^{k+1}, we have $Ax^{k+1} - b = (\mu^{k+1} - \mu^k)/\beta$. The first part of the assertion therefore follows from part (b). The second part follows from the fact that $w^\infty = (x^\infty, \mu^\infty)$ is a KKT point, see (a).

(d) The fourth assertion follows from [10] which is a recent improvement of a classical result stated in [5, Prop. 8].

(e), (f) Statement (e) is a standard result, however, for the sake of completeness and since it will be used to verify assertion (f), we include its proof here.

In view of (a), we may assume that $w^k \rightharpoonup w^\infty$ for some weak limit point $w^\infty = (x^\infty, \mu^\infty)$. Furthermore, using (b) and (c), we obtain

$$\begin{aligned} \|A_i x_i^{k+1} + \sum_{l \neq i} A_l x_l^k - b\| &\leq \|A_i(x_i^{k+1} - x_i^k)\| + \|\sum_{l=1}^N A_l x_l^k - b\| \\ &\leq \|A_i\| \|x_i^{k+1} - x_i^k\| + \|\sum_{l=1}^N A_l x_l^k - b\| \to 0. \end{aligned}$$

Since $\mathcal{X}$ is closed and convex, it is weakly sequentially closed, hence $x^\infty \in \mathcal{X}$. From $x^\infty, x^{k+1} \in \mathcal{X}$, (14), and the definition of the subdifferential, we obtain

$$f_i(x_i^\infty) - f_i(x_i^{k+1}) + \Big\langle A_i^* \mu^k + \beta A_i^* \big(A_i x_i^{k+1} + \sum_{l \neq i} A_l x_l^k - b\big) + \beta\gamma(x_i^{k+1} - x_i^k) \mid x_i^\infty - x_i^{k+1} \Big\rangle \geq 0.$$

Summation for $i = 1, \ldots, N$ yields, taking into account that $x^{k+1} - x^k \to 0$, $A_i x_i^{k+1} + \sum_{l \neq i} A_l x_l^k - b \to 0$, and the boundedness of $x^\infty - x^{k+1}$,

$$\begin{aligned} f(x^\infty) &\geq f(x^{k+1}) + \langle A^* \mu^k \mid x^{k+1} - x^\infty \rangle + \varepsilon^k \\ &= f(x^{k+1}) + \langle \mu^k \mid Ax^{k+1} - Ax^\infty \rangle + \varepsilon^k \\ &= f(x^{k+1}) + \langle \mu^k \mid Ax^{k+1} - b \rangle + \varepsilon^k \\ &= f(x^{k+1}) + \tilde{\varepsilon}^k, \end{aligned}$$

where $\varepsilon^k, \tilde{\varepsilon}^k$ are certain sequences converging to zero. Since f is lsc, it is also weakly sequentially lsc, cf. [2, Thm. 9.1], i.e. $\liminf_{k\to\infty} f(x^k) \geq f(x^\infty)$, and we therefore obtain

$$\limsup_{k\to\infty} f(x^{k+1}) \leq f(x^\infty) \leq \liminf_{k\to\infty} f(x^{k+1}).$$

Consequently, we have $f(x^k) \to f(x^\infty) = f^*$, the last equation holds because x^∞ is a minimizer of (1). This verifies statement (e). We next exploit that part to show assertion (f). To this end, first note that (e) together with the lsc property of all f_i implies

$$\begin{aligned} f(x^\infty) &= \sum_{l=1}^{N} f_l(x_l^\infty) = \sum_{l\neq i} f_l(x_l^\infty) + f_i(x_i^\infty) \\ &\leq \sum_{l\neq i} f_l(x_l^\infty) + \liminf_{k\to\infty} f_i(x_i^k) \leq \sum_{l=1}^{N} \liminf_{k\to\infty} f_l(x_l^k) \\ &\leq \liminf_{k\to\infty} \Big(\sum_{l=1}^{N} f_l(x_l^k) \Big) = \liminf_{k\to\infty} f(x^k) = \lim_{k\to\infty} f(x^k) = f(x^\infty), \end{aligned}$$

so that equality holds everywhere. In particular, it follows that $\liminf_{k\to\infty} f_i(x_i^k) = f_i(x_i^\infty)$. Together with (e), this further implies

$$\begin{aligned} \limsup_{k\to\infty} f_i(x^k) &\leq \limsup_{k\to\infty} \Big(\sum_{l=1}^{N} f_l(x_l^k) \Big) + \limsup_{k\to\infty} \Big(-\sum_{l\neq i} f_l(x_l^k) \Big) \\ &= f(x^\infty) - \sum_{l\neq i} \liminf_{k\to\infty} \Big(f_l(x_l^k) \Big) = f_i(x_i^\infty) = \liminf_{k\to\infty} f_i(x_i^k). \end{aligned}$$

This yields $f_i(x_i^k) \to f_i(x_i^\infty)$.

(g) By assertion (d) we have that $\operatorname{dist}(T(w^k), 0) \to 0$. Suppose that f_i is strongly convex. Then there exists a constant $\nu_i > 0$ such that

$$\begin{aligned} f_i(y_i) - f_i(x_i) &\geq \langle g_i(x_i) \mid y_i - x_i \rangle + \nu_i \|x_i - y_i\|^2, \\ f_i(x_i) - f_i(y_i) &\geq \langle g_i(y_i) \mid x_i - y_i \rangle + \nu_i \|x_i - y_i\|^2 \end{aligned}$$

for all $g_i(x_i) \in \partial f_i(x_i)$ and all $g_i(y_i) \in \partial f_i(y_i)$. Adding these inequalities yields

$$\langle g_i(x_i) - g_i(y_i) \mid x_i - y_i \rangle \geq 2\nu_i \|x_i - y_i\|^2. \tag{16}$$

Now let us take an element $v^k \in T(w^k) = T(x^k, \mu^k)$ such that $\|v^k - 0\| \leq \operatorname{dist}(0, T(w^k)) + 1/k$ for all $k \in \mathbb{N}$ which is always possible by definition of the distance function. Recalling the definition of the operator T and using the separability of the function f, we see that this v^k has a representation of the form

$$v^k =: \begin{pmatrix} g_1(x_1^k) \\ \vdots \\ g_i(x_i^k) \\ \vdots \\ g_N(x_N^k) \\ 0 \end{pmatrix} + \begin{pmatrix} A_1^* \mu^k \\ \vdots \\ A_i^* \mu^k \\ \vdots \\ A_N^* \mu^k \\ b - Ax^k \end{pmatrix} + \begin{pmatrix} s_1(x_1^k) \\ \vdots \\ s_i(x_i^k) \\ \vdots \\ s_N(x_N^k) \\ 0 \end{pmatrix}$$

for certain elements $s_i(x_i^k) \in N_{\mathcal{X}_i}(x_i^k)$ and $g_i(x_i^k) \in \partial f_i(x_i^k)$. In view of assertion (a), we also have $0 \in T(w^\infty) = T(x^\infty, \mu^\infty)$. Then we obtain from the monotonicity of the normal cone operators together with (16) that

$$\begin{aligned}
&\left\langle v^k - 0 \mid \begin{pmatrix} x^k - x^\infty \\ \mu^k - \mu^\infty \end{pmatrix} \right\rangle \\
&= \sum_{l=1}^N \langle g_l(x_l^k) - g_l(x_l^\infty) \mid x_l^k - x_l^\infty \rangle + \sum_{l=1}^N \langle A_l^* \mu^k - A_l^* \mu^\infty \mid x_l^k - x_l^\infty \rangle \\
&\quad + \sum_{l=1}^N \langle s_l(x_l^k) - s_l(x_l^\infty) \mid x_l^k - x_l^\infty \rangle + \langle (b - Ax^k) - (b - Ax^\infty) \mid \mu^k - \mu^\infty \rangle \\
&\geq \langle g_i(x_i^k) - g_i(x_i^\infty) \mid x_i^k - x_i^\infty \rangle + \langle Ax^\infty - Ax^k \mid \mu^k - \mu^\infty \rangle - \langle x^\infty - x^k \mid A^*\mu^k - A^*\mu^\infty \rangle \\
&\geq 2\nu_i \|x_i^k - x_i^\infty\|^2.
\end{aligned}$$

Since $\{v^k\}$ converges strongly to zero in view of (d), and $\{w^k\}$ is weakly convergent, the previous chain of inequalities shows that $x_i^k \to x_i^\infty$ (strongly). □

Remark 4.7. (a) As we have seen in Proposition 4.4, the operator $(I + \beta Q^{-1}T)^{-1}$ is firmly non-expansive in a suitable Hilbert space. By the Krasnoselsky-Mann iteration for firmly non-expansive operators, see [2, Cor. 5.17], we see that many statements of Theorem 4.6 remain true if we consider the more general iterative procedure

$$x^{k+1} := (1 - \rho^k)x^k + \rho^k \hat{x}^k, \quad \mu^{k+1} := (1 - \rho^k)\mu^k + \rho^k \hat{\mu}^k,$$

where $\hat{x}^k, \hat{\mu}^k$ denotes the outcome of one iteration of Algorithm 3.1 and $\rho^k \in [0, 2]$ satisfies $\sum_{k=1}^\infty \rho^k(1 - \rho^k) = \infty$. The choice $\rho^k = 1$ corresponds to our algorithm, whereas $\rho^k < 1$ and $\rho^k > 1$ are often called under- and overrelaxation, respectively. In view of our limited numerical experience, however, we do not benefit anything by taking $\rho^k \neq 1$.

(b) There exist several inexact versions of the proximal-point method in the literature, see for example [11, 13, 22, 28]. The previous analysis clearly shows that it is also possible to exploit these inexact proximal-point methods in order to obtain inexact counterparts of Algorithm 3.1. The corresponding details are left to the reader.

It is not difficult to see that the convergence theory in this section remains true as long as the bounded operator Q is self-adjoint and strongly monotone. Specifically, if Q from (12) comes from another splitting-type scheme, we obtain the same interpretation as a proximal-point method and therefore inherit its convergence properties. Unfortunately, Q being self-adjoint plays a central role here, as will be seen below in Example 4.8. This means that our convergence theory cannot be applied to the case where Algorithm 3.1 is replaced by a corresponding regularized Gauss-Seidel-type ADMM method because the resulting counterpart of the matrix M (hence also Q) from (11) would not be self-adjoint. But Q not being self-adjoint destroys all desired convergence properties. This is illustrated by the following counterexample in the finite-dimensional setting.

Example 4.8. In order to show that the proximal-point algorithm can only be applied when the operator Q is self-adjoint, let us define

$$T := \begin{pmatrix} 0 & -1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & 0 \end{pmatrix} \quad \text{and} \quad Q := \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}.$$

First we notice that T is maximally monotone, cf. [2, Cor. 20.28]. Furthermore, an easy calculation shows that

$$d^T Q d = \frac{1}{2}(d_1 + d_2)^2 + \frac{1}{2}(d_2 + d_3)^2 + \frac{1}{2}(d_1 + d_3)^2 > 0$$

for all $d \neq 0$, hence Q is positive definite and therefore yields a strongly monotone operator. However, the matrix Q is not symmetric, so we do not have a self-adjoint operator here. The proximal-point method is given by

$$x^{k+1} = \left(\begin{pmatrix} 0 & -1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & 0 \end{pmatrix} + \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \right)^{-1} \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} x^k = \begin{pmatrix} 1 & 1 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \end{pmatrix} x^k.$$

Further we see that

$$\begin{pmatrix} 1 & 1 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \end{pmatrix}^4 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

hence $x^{4k} = x^0$ for every $x^0 \in \mathbb{R}^3$ and all $k \in \mathbb{N}$. Thus the method does not converge.

5 A Strongly Convergent Algorithm

As shown in the previous section, the operator T from (6) is maximally monotone, hence our Jacobi-type ADMM method from Algorithm 3.1 yields weak convergence of the corresponding sequence $\{w^k\}$ since this method can be interpreted as a proximal-point method in a suitable Hilbert space. We also proved that the sequence x_i^k generated by Algorithm 3.1 converges strongly whenever f_i is strongly convex. But there are a lot of interesting functions that are not strongly convex.

On the other hand, it is often appreciated to have a strongly convergent algorithm at hand since the approximation of the numerical solution of the discretized problem to the actual solution of the non-discretized problem can be expected to be better. To this end, we note that there exist some modifications of the proximal-point method which are known to give strongly convergent iterates. One of these modifications is Halpern's method, see [16] for the original reference or the discussion in [2]. In order to describe the simple idea of Halpern's method, consider first the usual fix-point iteration

$$w^{k+1} = Jw^k \tag{17}$$

that converges weakly if J is firmly non-expansive and has at least one fixed point, see [2, Example 5.18]. If J is the resolvent of a maximally monotone operator, then it is firmly non-expansive and we obtain the proximal-point algorithm. In oder to be able to obtain weak convergence of only non-expansive operators J, Krasnoselsky and Mann blended a bit of the identity into (17) by using the iteration

$$w^{k+1} = \rho^k w^k + (1-\rho^k) J w^k, \tag{18}$$

where $\rho^k \in [0,1]$ and $\sum_{i=1}^{\infty} \rho^k(1-\rho^k) = +\infty$, see [2, Thm. 5.15]. Now the idea of Halpern was to replace the vector w^k from the identity map by a fixed vector w. Thus Halpern's iteration is $w^{k+1} = \rho^k w + (1-\rho^k) J w^k$, where the sequence $\{\rho^k\}$ satisfies the conditions

$$\rho^k \to 0, \quad \sum_{k=1}^{\infty} \rho^k = +\infty, \quad \sum_{k=1}^{\infty} |\rho^{k+1} - \rho^k| < \infty \tag{19}$$

which, in particular, hold for the choice $\rho^k := 1/k$. This method is known to be strongly convergent to the particular solution $\mathrm{Proj}_{\mathrm{Fix}\, J}\, w$.

Having in mind the proximal-point interpretation of our regularized Jacobi-type ADMM method from Algorithm 3.1 with $J = (I + \beta Q^{-1} T)^{-1}$, it is not surprising that the following modified method can be seen as Halpern's method.

Algorithm 5.1. *(Halpern-Regularized Jacobi-type ADMM Method)*

(S.0) Choose $(x^0, \mu^0), (x, \mu) \in \mathcal{X} \times \mathcal{K}$, *parameters* $\beta, \gamma > 0$, *set* $k := 0$ *and choose a sequence* $(\rho^k)_{k\in\mathbb{N}}$ *satisfying* (19).

(S.1) If a suitable termination criterion is satisfied: STOP.

(S.2) For $i = 1, \ldots, N$, *compute*

$$\tilde{x}_i^k := \operatorname*{arg\,min}_{x_i \in \mathcal{X}_i} \Big\{ f_i(x_i) + \langle \mu^k \mid A_i x_i \rangle + \frac{\beta}{2} \big(\|A_i x_i + \sum_{l \neq i} A_l x_l^k - b\|^2 + \gamma \|x_i - x_i^k\|^2 \big) \Big\} \tag{20}$$

(S.3) Define

$$\tilde{\mu}^k := \mu^k + \beta \Big(\sum_{l=1}^{N} A_l x_l^{k+1} - b \Big). \tag{21}$$

(S.4) Set

$$x^{k+1} := \rho^k x + (1-\rho^k)\tilde{x}^k, \qquad \mu^{k+1} := \rho^k \mu + (1-\rho^k)\tilde{\mu}^k.$$

(S.5) Set $k \leftarrow k+1$, *and go to (S.1).*

The global and strong convergence properties of the previous method follows immediately from known results about Halpern's modification of the standard proximal-point method and are summarized, for the sake of completeness, in the following result.

Theorem 5.2. *Suppose that Q is self-adjoint, strongly monotone and that there is at least one KKT point for the optimization problem* (1)*. Then the sequence $\{w^k\} = \{(x^k, \mu^k)\}$ generated by Algorithm 5.1 converges strongly to a KKT point $w^\infty = (x^\infty, \mu^\infty)$ of* (1)*.*

Proof. The operator $J = (Q + \beta T)^{-1}Q \overset{\text{Lem. } 4.3}{=} (I + \beta Q^{-1}T)^{-1}$ is non-expansive by Proposition 4.4, as a resolvent of a maximal monotone map in the Hilbert space $\mathcal{H} \times \mathcal{K}$ endowed with the scalar product $\langle \cdot \mid \cdot \rangle_Q$. It was shown in Proposition 4.4 that the iterates $\tilde{w}^k = (\tilde{x}^k, \tilde{\mu}^k)$ generated by (20) and (21) are equal to the iterates generated by $\tilde{w}^k := (I + \beta Q^{-1}T)^{-1}w^k$, where T is the maximal monotone operator defined in (6). Thus the iterates $w^{k+1} = (x^{k+1}, \mu^{k+1})$ generated by Algorithm 5.1 are equal to the iterates generated by

$$w^{k+1} = \rho^k w + (1 - \rho^k)\tilde{w}^k \;=\; \rho^k w + (1 - \rho^k)(I + \beta Q^{-1}T)^{-1}w^k,$$

where w is a fixed vector and ρ^k is given as in Algorithm 5.1. But this is the usual Halpern-type iteration for the non-expansive operator $(I + \beta Q^{-1}T)^{-1}$. The assertion now follows from the convergence properties of Halpern's iteration, cf. [2, Thm. 30.1]. □

The strong convergence of the iterates w^k to a KKT point of the optimization problem (1) immediately implies that all the other statements (if not superfluous) known from Theorem 4.6 automatically also hold for Algorithm 5.1.

6 Application to Domain Decomposition

Domain decomposition is a technique for the solution of boundary value problems which splits the given domain into smaller ones. Prominent examples are the methods by Schwarz which, for certain problems, were shown to be equivalent to an augmented Lagrangian method applied to the corresponding optimization problem, see [14, 25]. Here we follow a similar idea and show how our regularized Jacobi-type ADMM methods can be used to obtain suitable domain decomposition methods. The central idea is described in Section 6.1. An explicit realization of our methods applied to a particular instance is discussed in Section 6.2, whereas we compute a suitable lower bound for the choice of our parameter γ in Section 6.3.

6.1 Non-Overlapping Domain Decomposition

In this subsection we will follow [1] to decompose the domain of a partial differential equation (PDE). For example, Figure 1 shows how the unit square can be decomposed in four different squares. Although we will describe the idea of the domain decomposition method in a general context, we will often refer to this particular example for its explicit realization. This way, we are able to avoid some technical notation; moreover, the central idea can be explained much better for this special setting.

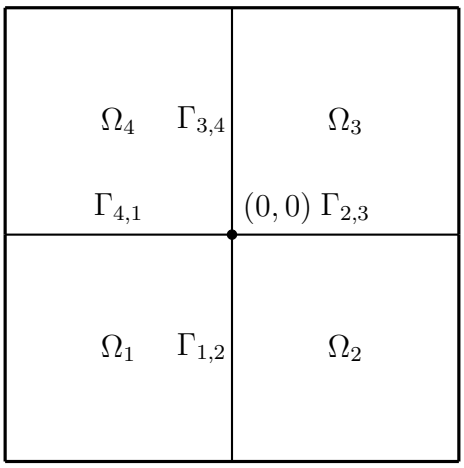

Figure 1: The unit square Ω decomposed in four squares Ω_i

Let us consider the problem of solving the Laplace equation

$$-\Delta y(x) = u(x) \quad \forall x \in \Omega, \qquad y(x) = 0 \quad \forall x \in \partial\Omega,$$

where Ω is a bounded convex Lipschitz domain in $\mathbb{R}^d$ and $u \in L^2(\Omega)$. Then the associated weak formulation is

$$\langle \nabla y \mid \nabla v \rangle_{L^2(\Omega)} = \langle u \mid v \rangle_{L^2(\Omega)} \qquad \forall v \in H_0^1(\Omega). \tag{22}$$

Existence of a weak solution follows from the Lax-Milgram Theorem. Moreover, it is well known that solving (22) is equivalent to finding a solution of the optimization problem

$$\min_{y \in H_0^1(\Omega)} \big\{ \frac{1}{2} \langle \nabla y \mid \nabla y \rangle_{L^2(\Omega)} - \langle u \mid y \rangle_{L^2(\Omega)} \big\}. \tag{23}$$

Now let us decompose our domain Ω into disjoint convex Lipschitz subdomains Ω_i such that $\Omega = \Omega_1 \dot\cup \ldots \dot\cup \Omega_N$. We equip H_0^1 with the standard H^1-Norm, in order to allow subdomains that are in the interior of Ω, which is not possible in the methods from [1, 25].

Unfortunately, in order to describe the main idea in a concise way, there is a technical overhead; in particular, we need to define some appropriate index sets. To this end, let J denote the set of all pairs (i, j) such that $\partial\Omega_i \cap \partial\Omega_j$ does not have vanishing $d-1$ dimensional measure, thus in Figure 1 we have $J = \{(1,2),(2,3),(3,4),(4,1),(2,1),(3,2),(4,2),(1,4)\}$, whereas, e.g., the pairs $(1,3)$ and $(2,4)$ do not belong to J. Further we want to denote with J^o a subset of J that does not contain permutations, thus we choose $J^o = \{(1,2),(2,3),(3,4),(4,1)\}$, the small index "o" stands for ordered. Now let us define $\Gamma_{i,j} = \partial\Omega_i \cap \partial\Omega_j$ for all $(i,j) \in J$ and $\Gamma_i = \partial\Omega \cap \partial\Omega_i$. We denote with $H^1_{\Gamma_i}(\Omega_i)$ the $H^1(\Omega_i)$ functions with trace equal to zero in Γ_i.

In the following, we distinguish between the trace operators $\text{trace}_{\Gamma_{i,j}}$ and $\text{trace}_{\Gamma_{j,i}}$: The former is defined on the sub-domain Ω_i with assigned values on the boundary $\Gamma_{i,j}$, whereas the latter denotes the trace operator defined on the sub-domain Ω_j with assigned values on the (same) boundary $\Gamma_{j,i} = \Gamma_{i,j}$.

Remark 6.1. By partial integration, it can be seen easily that

$$y \in H_0^1(\Omega) \Longleftrightarrow \big(y_i \in H^1_{\Gamma_i}(\Omega_i)\, \forall i = 1, \dots, N \text{ and } \operatorname{trace}_{\Gamma_{i,j}}(y_i) = \operatorname{trace}_{\Gamma_{j,i}}(y_j)\, \forall (i,j) \in J^o\big);$$

furthermore, it holds that $u \in L^2(\Omega)$ if and only if $u_i \in L^2(\Omega_i)$ for all $i = 1, \dots, N$, where u_i denotes the restriction of the given mapping u on Ω_i.

Hence (23) can be written as

$$\begin{aligned} \min_{\substack{y_i \in H^1_{\Gamma_i}(\Omega_i) \\ i=1,\dots,N}} & \Big\{ \sum_{i=1}^{N} \big(\frac{1}{2} \langle \nabla y_i \mid \nabla y_i \rangle_{L^2(\Omega_i)} - \langle u_i \mid y_i \rangle_{L^2(\Omega_i)} \big) \Big\} \\ \text{s.t. } & \operatorname{trace}_{\Gamma_{i,j}}(y_i) = \operatorname{trace}_{\Gamma_{j,i}}(y_j) \qquad \forall (i,j) \in J^o. \end{aligned} \tag{24}$$

Since the trace operator is linear and continuous, this optimization problem is exactly of the form (1), so we can apply Algorithms 3.1 or 5.1 to (24). Using the notation from (1), we have

- $\mathcal{X}_i = \mathcal{H}_i = H^1_{\Gamma_i}(\Omega_i)$
- $\mathcal{K} = \prod_{(i,j) \in J^o} L^2(\Gamma_{i,j})$
- $f_i : \mathcal{X}_i \to \mathbb{R}$ by $f_i(y_i) := \frac{1}{2} \langle \nabla y_i \mid \nabla y_i \rangle_{L^2(\Omega_i)} - \langle u_i \mid y_i \rangle_{L^2(\Omega_i)}$
- $A_i : \mathcal{X}_i \to \mathcal{K}$ correspond to the linear trace constraints from (24).

The latter is somewhat technical to describe in general, but for the particular domain from Figure 1, it is easy to see that the corresponding A_i's are given by

$$A_1 = \begin{pmatrix} \operatorname{trace}_{\Gamma_{1,2}} \\ 0 \\ 0 \\ -\operatorname{trace}_{\Gamma_{1,4}} \end{pmatrix}, A_2 = \begin{pmatrix} -\operatorname{trace}_{\Gamma_{2,1}} \\ \operatorname{trace}_{\Gamma_{2,3}} \\ 0 \\ 0 \end{pmatrix}, A_3 = \begin{pmatrix} 0 \\ -\operatorname{trace}_{\Gamma_{3,2}} \\ \operatorname{trace}_{\Gamma_{3,4}} \\ 0 \end{pmatrix}, A_4 = \begin{pmatrix} 0 \\ 0 \\ -\operatorname{trace}_{\Gamma_{4,3}} \\ \operatorname{trace}_{\Gamma_{4,1}} \end{pmatrix}. \tag{25}$$

We will later need these definitions to estimate the proximal constant γ. Further it is easy to see that the subproblems in Algorithms 3.1 and 5.1 consist of solving a PDE on the subdomains Ω_i in every step, or, in a finite element context, we have to solve N lower-dimensional linear system of equations. The corresponding details for the particular domain from Figure 1 will be given in the next section.

Remark 6.2. If $\partial\Omega_i \cap \partial\Omega$ has $(d-1)$-dimensional measure larger than zero for all i, then the Poincaré-inequality implies that all functions f_i are strongly convex. Consequently, the corresponding iterates generated by Algorithm 3.1 applied to the domain decomposition context converge strongly in view of Theorem 4.6 (g).

6.2 Application of Algorithms

Now we want to apply Algorithm 3.1 to problem (24) for the domain displayed in Figure 1. Similar considerations hold for the application of Algorithm 5.1, the corresponding details are left to the reader.

The constraints in (24) are equivalent to $\operatorname{trace}_{\Gamma_{i,j}} y_i - \operatorname{trace}_{\Gamma_{j,i}} y_j = 0$. Following standard notation in the field of applied analysis, we will omit the trace operator and identify y_i on the boundary with its trace, thus the corresponding constraint gets $y_i - y_j = 0$ in $\Gamma_{i,j}$. Consequently, the subproblems resulting from the four domains read as follows:

$$y_1^{k+1} = \operatorname*{arg\,min}_{y_1 \in H^1_{\Gamma_1}(\Omega_1)} \left\{ \left(\frac{1}{2} \langle \nabla y_1 \mid \nabla y_1 \rangle_{L^2(\Omega_1)} - \langle u_1 \mid y_1 \rangle_{L^2(\Omega_1)} \right) + \langle \mu_{1,2}^k \mid y_1 \rangle_{L^2(\Gamma_{1,2})} - \langle \mu_{4,1}^k \mid y_1 \rangle_{L^2(\Gamma_{4,1})} \right.$$
$$\left. + \frac{\beta}{2} \left(\|y_1 - y_2^k\|^2_{L^2(\Gamma_{1,2})} + \|y_1 - y_4^k\|^2_{L^2(\Gamma_{4,1})} \right) + \frac{\beta\gamma}{2} \|y_1 - y_1^k\|^2_{H^1_{\Gamma_1}(\Omega_1)} \right\}$$

$$y_2^{k+1} = \operatorname*{arg\,min}_{y_2 \in H^1_{\Gamma_2}(\Omega_2)} \left\{ \left(\frac{1}{2} \langle \nabla y_2 \mid \nabla y_2 \rangle_{L^2(\Omega_2)} - \langle u_2 \mid y_2 \rangle_{L^2(\Omega_2)} \right) + \langle \mu_{2,3}^k \mid y_2 \rangle_{L^2(\Gamma_{2,3})} - \langle \mu_{1,2}^k \mid y_2 \rangle_{L^2(\Gamma_{1,2})} \right.$$
$$\left. + \frac{\beta}{2} \left(\|y_2 - y_1^k\|^2_{L^2(\Gamma_{1,2})} + \|y_2 - y_3^k\|^2_{L^2(\Gamma_{2,3})} \right) + \frac{\beta\gamma}{2} \|y_3 - y_3^k\|^2_{H^1_{\Gamma_2}(\Omega_2)} \right\}$$

$$y_3^{k+1} = \operatorname*{arg\,min}_{y_3 \in H^1_{\Gamma_3}(\Omega_3)} \left\{ \left(\frac{1}{2} \langle \nabla y_3 \mid \nabla y_3 \rangle_{L^2(\Omega_3)} - \langle u_3 \mid y_3 \rangle_{L^2(\Omega_3)} \right) + \langle \mu_{3,4}^k \mid y_3 \rangle_{L^2(\Gamma_{3,4})} - \langle \mu_{2,3}^k \mid y_3 \rangle_{L^2(\Gamma_{2,3})} \right.$$
$$\left. + \frac{\beta}{2} \left(\|y_3 - y_2^k\|^2_{L^2(\Gamma_{2,3})} + \|y_3 - y_4^k\|^2_{L^2(\Gamma_{3,4})} \right) + \frac{\beta\gamma}{2} \|y_3 - y_3^k\|^2_{H^1_{\Gamma_3}(\Omega_3)} \right\}$$

$$y_4^{k+1} = \operatorname*{arg\,min}_{y_4 \in H^1_{\Gamma_4}(\Omega_4)} \left\{ \left(\frac{1}{2} \langle \nabla y_4 \mid \nabla y_4 \rangle_{L^2(\Omega_4)} - \langle u_4 \mid y_4 \rangle_{L^2(\Omega_4)} \right) + \langle \mu_{4,1}^k \mid y_4 \rangle_{L^2(\Gamma_{4,1})} - \langle \mu_{3,4}^k \mid y_4 \rangle_{L^2(\Gamma_{3,4})} \right.$$
$$\left. + \frac{\beta}{2} \left(\|y_4 - y_3^k\|^2_{L^2(\Gamma_{3,4})} + \|y_4 - y_1^k\|^2_{L^2(\Gamma_{4,1})} \right) + \frac{\beta\gamma}{2} \|y_3 - y_3^k\|^2_{H^1_{\Gamma_4}(\Omega_4)} \right\},$$

and the multiplier update is

$$\begin{aligned} \mu_{1,2}^{k+1} &= \mu_{1,2}^k + \beta(y_1^{k+1} - y_2^{k+1}), & \mu_{2,3}^{k+1} &= \mu_{2,3}^k + \beta(y_2^{k+1} - y_3^{k+1}), \\ \mu_{3,4}^{k+1} &= \mu_{3,4}^k + \beta(y_3^{k+1} - y_4^{k+1}), & \mu_{4,1}^{k+1} &= \mu_{4,1}^k + \beta(y_4^{k+1} - y_1^{k+1}), \end{aligned}$$

where the equalities hold in the boundary spaces $L^2(\Gamma_{i,j})$ for $(i,j) \in J^o$.

Using the sign vector defined by

$$\alpha_{i,j} = \begin{cases} +1 & \text{if } (i,j) \in J^o = \{(1,2),(2,3),(3,4),(4,1)\} \\ -1 & \text{if } (i,j) \in J \setminus J^o = \{(2,1),(3,2),(4,3),(1,4)\} \end{cases},$$

we can rewrite these subproblems more compactly as

$$y_i^{k+1} = \operatorname*{arg\,min}_{y_i \in H^1_{\Gamma_i}(\Omega_i)} \left\{ \left(\frac{1}{2}\langle \nabla y_i \mid \nabla y_i \rangle_{L^2(\Omega_i)} - \langle u_i \mid y_i \rangle_{L^2(\Omega_i)} \right) + \sum_{j:(i,j)\in J} \alpha_{i,j} \langle \mu_{i,j}^k \mid y_i \rangle_{L^2(\Gamma_{i,j})} \right.$$
$$\left. + \frac{\beta}{2} \sum_{j:(i,j)\in J} \|y_i - y_j^k\|^2_{L^2(\Gamma_{i,j})} + \frac{\beta\gamma}{2} \|y_i - y_i^k\|^2_{H^1_{\Gamma_i}(\Omega_i)} \right\} \tag{26}$$

for all $i = 1, \ldots, 4$. The multiplier update gets

$$\mu_{i,j}^{k+1} = \mu_{i,j}^k + \beta(\alpha_{i,j} y_i^{k+1} + \alpha_{j,i} y_i^{k+1}) \qquad \forall (i,j) \in J^o \qquad \mu_{i,j}^{k+1} = \mu_{j,i}^{k+1} \tag{27}$$

in $L^2(\Gamma_{i,j})$.

The optimality conditions of (26) are necessary and sufficient since the subproblems are strongly convex. These optimality conditions are given by

$$\langle \nabla y_i^{k+1}, \nabla v_i \rangle_{L^2(\Omega_i)} + \sum_{j:(i,j)\in J} \alpha_{i,j} \langle \mu_{i,j}^k \mid v_i \rangle_{L^2(\Gamma_{i,j})} + \beta \sum_{j:(i,j)\in J} \langle y_i^{k+1} - y_j^k \mid v_i \rangle_{L^2(\Gamma_{i,j})}$$
$$+ \beta\gamma \langle y_i^{k+1} - y_i^k \mid v_i \rangle_{L^2(\Omega_i)} + \beta\gamma \langle \nabla y_i^{k+1} - \nabla y_i^k \mid \nabla v_i \rangle_{L^2(\Omega_i)} = \langle u_i \mid v_i \rangle_{L^2(\Omega_i)} \tag{28}$$

for all $v_i \in H^1_{\Gamma_i}(\Omega_i)$, $i = 1, \ldots, 4$. This is the weak formulation of the PDE

$$\begin{aligned} -(1+\beta\gamma)\triangle y_i + \beta\gamma y_i &= u + \beta\gamma y_i^k - \beta\gamma \triangle y_i^k && \text{in } \Omega_i \\ \beta y_i + (1+\beta\gamma)\tfrac{\partial y_i}{n_i} &= \beta y_j^k + \beta\gamma \tfrac{\partial y_i^k}{n_i} - \alpha_{i,j}\mu_{i,j}^k && \text{in } \Gamma_{i,j}\ \forall j : (i,j) \in J \\ y_i &= 0 && \text{in } \partial\Omega \cap \partial\Omega_i, \end{aligned} \tag{29}$$

where n_i denotes the outer normal of Ω_i, $i = 1, \ldots, 4$.

Thus Algorithms 3.1 and 5.1 basically consist of solving the (uniquely determined) PDE (28) or (29), respectively, and thereafter doing the μ-update (27).

6.3 Estimating the Proximal Constant γ

Now we want to figure out how to choose the constant γ in the Algorithms 3.1 and 5.1 for the problem (24) on the domain from Figure 1. Thus we need to estimate the operator norm of M, defined in (11). To this end, we first state a lemma that estimates the operator norm of the trace operator to a boundary part.

Lemma 6.3. *Suppose $\Omega_i \subset \mathbb{R}^2$ is a rectangle with side length L_1 and L_2, i.e. only through rotation and translation it can be taken to a form $(0, L_1) \times (0, L_2)$, suppose that Γ is a side of Ω_i with length L_2, then $\| \operatorname{trace}_\Gamma \|^2 \leq 2 \max\{L_1, \frac{1}{L_1}\}$.*

Proof. To prove this lemma, we will follow [24, Thm. A.4]. Now first suppose that $v \in C^1([0,L])$, hence $v(x) = v(y) + \int_y^x v'(s)ds$ and therefore

$$|v(x)| \leq |v(y)| + \int_0^L |v'(s)|ds \leq |v(y)| + L^{\frac{1}{2}}\|v'\|_{L^2(0,L)} \quad \forall x, y \in [0,L].$$

Squaring both sides, integrating with respect to y, and using Young's inequality, we obtain

$$Lv(x)^2 \leq 2\|v\|^2_{L^2(0,L)} + 2L^2\|v'\|^2_{L^2(0,L)}$$

or, equivalently,

$$v(x)^2 \leq \frac{2}{L}\|v\|^2_{L^2(0,L)} + 2L\|v'\|^2_{L^2(0,L)}. \tag{30}$$

Now suppose that $y \in C^1(\overline{\Omega}_i)$ and w.l.o.g. let $\Gamma = \{0\} \times (0, L_2) \subset \partial\Omega_i$ be one boundary of Ω_i. We obtain by (30) that

$$y(0,x_2)^2 \leq \frac{2}{L_1}\int_0^{L_1} y(x_1,x_2)^2 dx_1 + 2L_1 \int_0^{L_1} \partial_1 y(x_1,x_2)^2 dx_1.$$

Integrating this equation with respect to x_2, we obtain

$$\begin{aligned}\|\operatorname{trace}_\Gamma y\|^2_{L^2(\Gamma)} = \|y\|^2_{L^2(\Gamma)} &= \int_0^{L_2} y(0,x_2)^2 dx_2 \\ &\leq \frac{2}{L_1}\|y\|^2_{L^2(\Omega_i)} + 2L_1\|\nabla y\|^2_{L^2(\Omega_i)} \leq 2\max\{L_1, \frac{1}{L_1}\}\|y\|^2_{H^1(\Omega_i)}.\end{aligned}$$

The claim follows from the density of $C^1(\overline{\Omega}_i)$ in $H^1(\Omega_i)$ and that rotation and translation do not change the operator norm. □

Lemma 6.4. *For our example domain displayed in Figure 1 we obtain $\|M\| < 5.7$, where M is defined as in* (11).

Proof. We see from (25) and $\langle A_i^* A_j x_j \mid y_i\rangle = \langle A_j x_j \mid A_i y_i\rangle$ that

$$\begin{aligned} A_1^*A_2 &= -\operatorname{trace}^*_{\Gamma_{1,2}} \operatorname{trace}_{\Gamma_{2,1}}, \quad A_2^*A_1 = -\operatorname{trace}^*_{\Gamma_{2,1}} \operatorname{trace}_{\Gamma_{1,2}} \\ A_1^*A_4 &= -\operatorname{trace}^*_{\Gamma_{1,4}} \operatorname{trace}_{\Gamma_{4,1}}, \quad A_4^*A_1 = -\operatorname{trace}^*_{\Gamma_{4,1}} \operatorname{trace}_{\Gamma_{1,4}} \\ A_2^*A_3 &= -\operatorname{trace}^*_{\Gamma_{2,3}} \operatorname{trace}_{\Gamma_{3,2}}, \quad A_3^*A_2 = -\operatorname{trace}^*_{\Gamma_{3,2}} \operatorname{trace}_{\Gamma_{2,3}}, \\ A_3^*A_4 &= -\operatorname{trace}^*_{\Gamma_{3,4}} \operatorname{trace}_{\Gamma_{4,3}}, \quad A_3^*A_4 = -\operatorname{trace}^*_{\Gamma_{3,4}} \operatorname{trace}_{\Gamma_{4,3}}, \\ A_1^*A_3 &= 0, \quad A_3^*A_1 = 0, \quad A_2^*A_4 = 0, \quad A_4^*A_2 = 0. \end{aligned}$$

With this we further notice that

$$\|Mx\|^2 = \left\| \left(\sum_{\substack{l=1 \\ l\neq i}}^4 A_i^* A_l x_l \right)_{i=1}^4 \right\|^2_{\mathcal{H}}$$

$$
\begin{aligned}
&\leq \|A_2^* A_1 x_1\|^2 + \|A_4^* A_1 x_1\|^2 + \|A_1^* A_2 x_2\|^2 + \|A_3^* A_2 x_2\|^2 \\
&\quad + \|A_2^* A_3 x_3\|^2 + \|A_4^* A_3 x_3\|^2 + \|A_1^* A_4 x_4\|^2 + \|A_3^* A_4 x_4\|^2 \\
&\leq 8 \cdot \max_{(i,j)\in J^o} \{\| \operatorname{trace}_{\Gamma_{i,j}} \|^2\} \|x\|^2,
\end{aligned}
$$

hence with the last lemma and $L = 0.5$, we obtain $\|M\| \leq \sqrt{8 \cdot 4} = \sqrt{32} < 5.7$. □

The previous result gives us an estimate of the constant γ appearing in Algorithms 3.1 and 5.1, since it is required that $\gamma > \|M\|$.

7 Numerical Results

In this section, we want to illustrate the numerical behavior of Algorithm 3.1. To do so, we first discuss the behavior of the domain decomposition technique described in Section 6. Afterwards we compare Algorithm 3.1 to some related ones from the literature using the (sparse) l^1-minimization problem.

7.1 Domain Decomposition

We implemented the domain decomposition algorithm described in Subsection 6.1 with Python and the FEniCS program package, version 2017.1, see `https://fenicsproject.org/`. We used the test example

$$
-\Delta y = -6 \quad \text{in } \Omega, \qquad y(x) = 1 + x_1^2 + 2x_2^2 \quad \forall x \in \partial\Omega, \tag{31}
$$

whose exact solution is given by $y(x) = 1 + x_1^2 + 2x_2^2$, cf. [23]. The theory from Section 6.1 applies with standard arguments also to arbitrary Dirichlet conditions in $H^{1/2}(\partial\Omega)$, thus (31) is covered by our theory. Motivated by the discussion in Section 6.3, we chose $\gamma = 5.7$ and $\beta = 1$ as the parameters in our explicit implementation of Algorithm 3.1. As a termination criterion we used $\|y_i^{k+1} - y_i^k\|^2_{L^2(\Omega_i)} \leq \varepsilon$ and $\|y_i^{k+1} - y_j^{k+1}\|_{L^2(\Gamma_{i,j})} \leq \varepsilon$, where the L^2-norm is the approximate L^2-norm provided by FEniCS. We are aware that in the first criterion the H^1-norm would be better but this norm is quite difficult to compute in the finite element context.

Table 1: Some results of the regularized Jacobi-type ADMM method from Algorithm 3.1 with different choices of the termination parameter ε and different mesh sizes for each fixed ε.

ε	largest edge in mesh	number of iterations =: k	$\|y^k - y^{exact}\|_{L^2(\Omega)}$
0.01	0.042	28	$2.5 \cdot 10^{-3}$
0.01	0.025	28	$2.3 \cdot 10^{-3}$
0.01	0.013	28	$2.3 \cdot 10^{-3}$
0.01	0.0042	28	$2.3 \cdot 10^{-3}$
0.01	0.0013	28	$2.3 \cdot 10^{-3}$
0.001	0.042	63	$4.2 \cdot 10^{-4}$
0.001	0.025	62	$1.8 \cdot 10^{-4}$
0.001	0.013	63	$1.1 \cdot 10^{-4}$
0.001	0.0042	64	$9.4 \cdot 10^{-5}$
0.001	0.0013	64	$9.3 \cdot 10^{-5}$
0.0001	0.042	289	$3.9 \cdot 10^{-4}$
0.0001	0.025	250	$1.2 \cdot 10^{-4}$
0.0001	0.013	284	$3.2 \cdot 10^{-5}$
0.0001	0.0042	302	$6.6 \cdot 10^{-6}$
0.0001	0.0013	308	$3.9 \cdot 10^{-6}$

We made some experiments with different mesh sizes and different ε, the corresponding results are summarized in Table 1. The results indicate that the number of iterations is (almost) independent of the mesh size. Moreover, taking into account the dimension of the discretized problem, the number of iterations is relatively small for all test problem instances. Finally, the last column in Table 1 shows that the exact error (not used as a termination criterion in our immplementation since usually the exact solution is unknown) is surprisingly small for a method whose local rate of convergence is (in general) sublinear.

To visualize the solution process, we also present some pictures with approximate solutions generated by the regularized Jacobi-type ADMM method from Algorithm 3.1, see Figures 2–4. These figures correspond to three different choices of ε and present the computed solution for a mesh size whose biggest edge length was always the same and around 0.013.

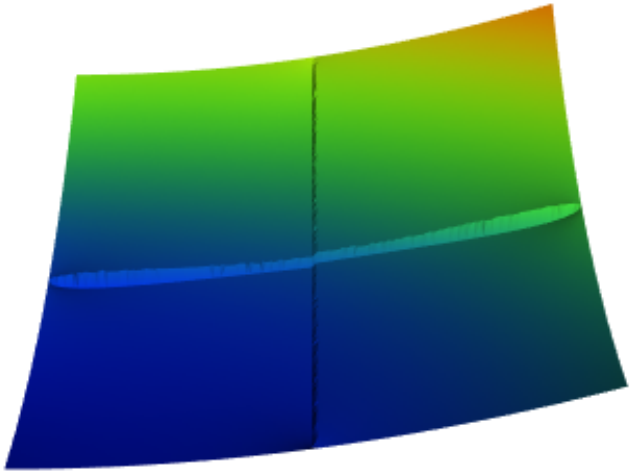

Figure 2: Plot of solution with $\varepsilon = 0.1$, number of iterations 5 , $\|y^5 - y^{exact}\|_{L^2(\Omega)} = 0.047$, there are strong edges between the solutions on the subdomains.

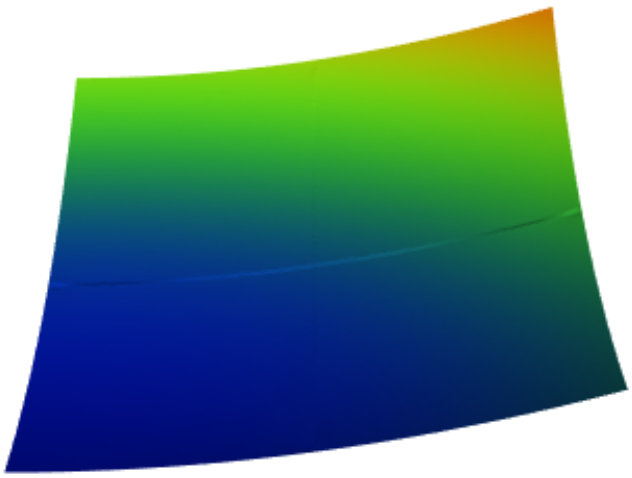

Figure 3: Plot of solution with $\varepsilon = 0.01$, number of iterations 28 , $\|y^{28} - y^{exact}\|_{L^2(\Omega)} = 0.0023$, the edges between the solutions on the subdomains are still clearly visible.

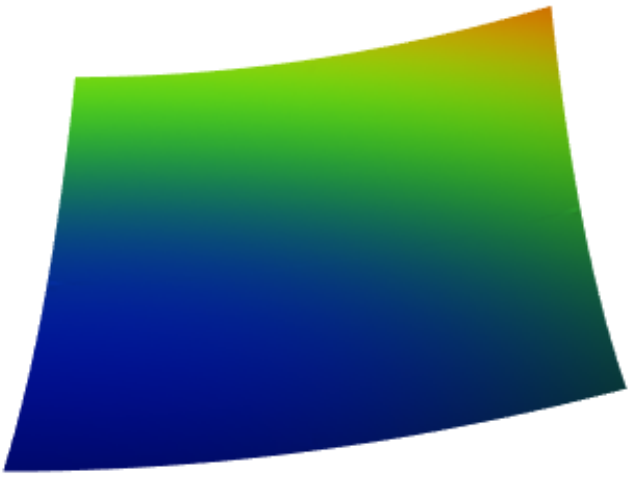

Figure 4: Plot of solution with $\varepsilon = 0.001$, number of iterations 63 , $\|y^{63} - y^{exact}\|_{L^2(\Omega)} = 1.1 \cdot 10^{-4}$. If we zoom in we can still see a small inaccuracy in the middle of the right edge. This inaccuracy will be still there with a finer grid, but disappears when ε is choosen to be smaller.

7.2 l^1 Minimization

One of the most used test problems for separable convex algorithms is the class of l^1 minimization problems. Among this class of problems, we chose the basis pursuit problem to compare the different Jacobi-type ADMM methods outlined in Remark 3.2 to each other. Hence we consider the optimization problem

$$\min \|x\|_1 \quad \text{s.t.} \quad Ax = b,$$

where $A \in \mathbb{R}^{m \times n}$. Thus we are able to split our problem in n one-dimensional problems, whose solutions can be computed analytically, see [17, Section 7.4.1.].

We use the technique of performance profiles for benchmarking optimization algorithms as introduced in [8]. Let us explain this technique a bit: We have a set of test problems P and apply different solvers from a set of solvers S to them. The number of iterations that the solver $s \in S$ needs for the problem $p \in P$ will be denoted by $t_{p,s}$, if the solver $s \in S$ does not solve the problem after a maximal iteration count, set $t_{p,s} = \infty$. Define the *performance ratio* of solver s to the problem p by

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s} \mid s \in S\}}.$$

The *performance ratio* of the solver $s \in S$ is now

$$\rho_s(\tau) = \frac{1}{|P|} \operatorname{size}\Big\{p \in P \mid r_{p,s} \leq \tau\Big\}.$$

That means $\rho_s(\tau)$ describes the number of test problems that the method s solves with a maximum of $\tau \cdot \min\{t_{p,s} \mid s \in S\}$ iterations.

The test problem set that was used in our numerical test is the SPEAR collection from `http://wwwopt.mathematik.tu-darmstadt.de/spear/` that provides us also with the exact solution x_{exact}. We only used the test problems whose number of columns was smaller than 3000. As a termination criterion we took $\|x^k - x_{exact}\|_\infty \leq 10^{-4}$ and the problem was considered not solved if the algorithm required more than one million iterations.

Our comparison includes the following algorithms:

1. The regularized Jacobi-type ADMM method from Algorithm 3.1 with parameters $\beta = 0.002$ and $\gamma = 1.1 \cdot \|A^T A - \mathrm{diag}(A^T A)\|_\infty$, where $\|A\|_\infty$ denotes the maximum absolute row sum of the current matrix A from the test problem set.

2. The $A_i^T A_i$-norm regularized Jacobi-type ADMM as described in Remark 3.2 (d) using the parameter $\beta = 003$.

3. The Jacobi-type ADMM as described in Remark 3.2 (a), with step size $\alpha = 1.999 \cdot (1 - \sqrt{\frac{N}{N+1}})$ as suggested in [17], where N denotes the number of columns of the matrix A. The penalty parameter β was chosen to be $\beta = 0.2$.

4. The twisted ADMM described mentioned in Remark 3.2 (e), with penalty parameter $\beta = 0.0001$ and proximal constant $\gamma = 1.1 \cdot (\max\{\mathrm{diag}(A^T A)\} - \min\{\mathrm{diag}(A^T A)\})$, as suggested in [29].

5. The regularized Jacobi-type ADMM method from [7] that is equal to the one from Algorithm 3.1 except for the choice of the proximal constant γ and a step size τ in the dual variable, as already discussed in Remark 3.2 (c). We choose the parameters $\beta = 0.003$, $\tau = 0.7$ and $\gamma_i = 1.1 \cdot (\frac{N}{2-\tau} - 1) A_i^T A_i$.

6. The Jacobi-type ADMM as described in Remark 3.2 (a), but this time with the step size strategy

$$\alpha = 1 \cdot \frac{\|w^k - \hat{w}^k\|_G^2 + 2(\mu^k - \hat{\mu}^k)^T (A(x^k - \hat{x}^k))}{\|w^k - \hat{w}^k\|_G^2},$$

   introduced in [17] and the penalty parameter $\beta = 0.06$.

The above choices of the parameters are either motivated by the corresponding theory or based on some preliminary numerical experiments to get an optimal behavior for each of the algorithms investigated here.

The amount of work per iteration for the first five methods is essentially the same. Hence the performance profile presented in Figure 5 based on the iteration count gives a good idea of the relative performance of each of these methods. The reason for using the iteration count and not the computation time is that we implemented the algorithms in MATLAB and CPU times provided by MATLAB seem to somewhat unreliable.

In Figure 5 the Algorithm 3.1 has by far the best performance among all Jacobi-type ADMM methods considered here. The criteria for the choice of γ in the twisted ADMM

method from [29] and the regularized Jacobi ADMM from [7] seem to be more restrictive and therefore lead to slower convergence of the corresponding algorithms. Furthermore, since all test problems have a relatively high dimension with $N \geq 1024$, it follows that the regularization method involving the $A_i^T A_i$ term yields a very high proximal constant γ which leads to the poor behaviour of this method; this disadvantage may vanish for problems with smaller dimensions. The Jacobi ADMM with constant step size has such a poor numerical behavior, since it has only a very small step size when the number of subproblems is high.

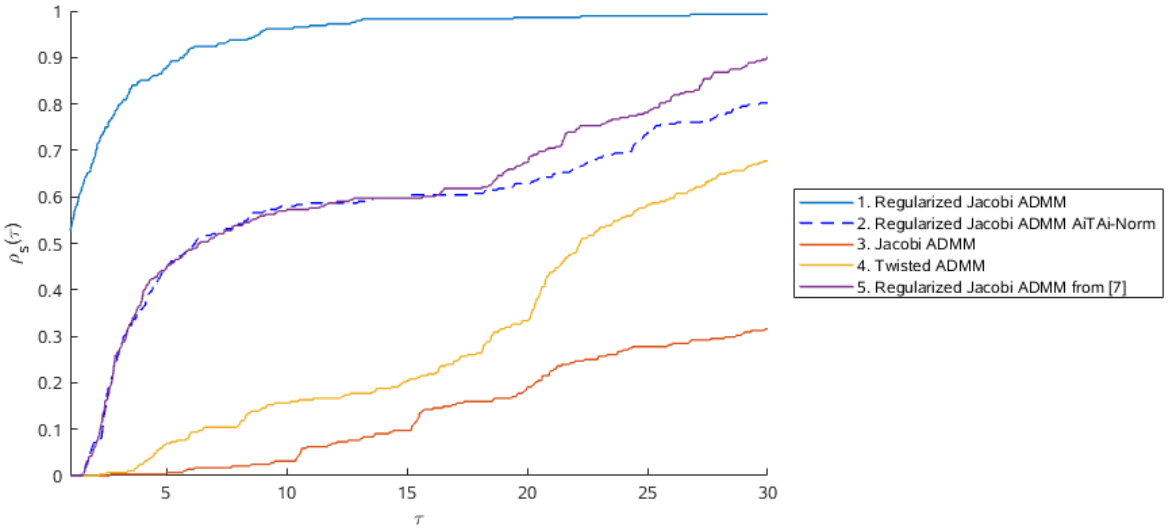

Figure 5: Performance profile for the first five Jacobi-type ADMM methods.

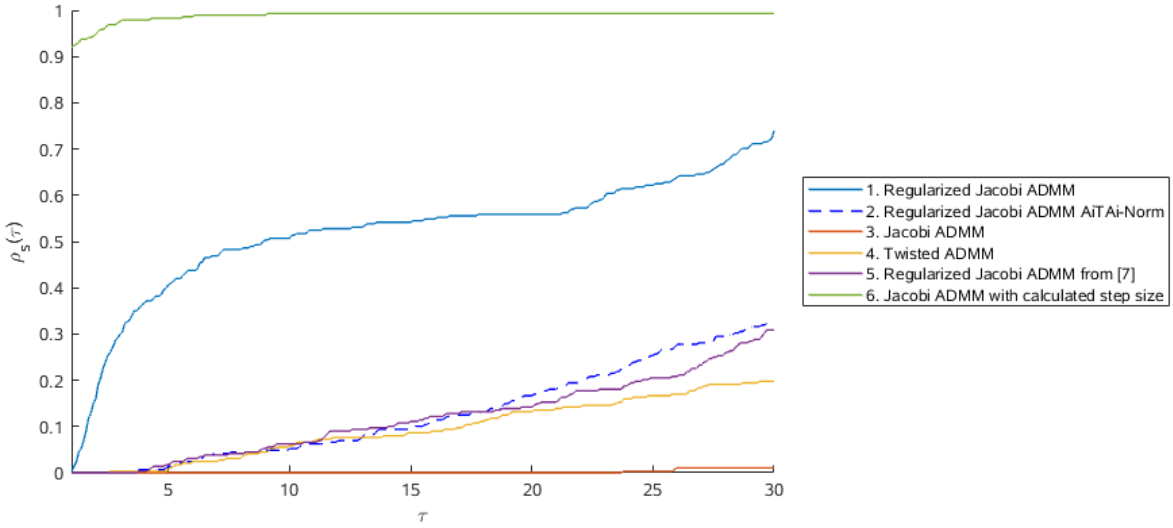

Figure 6: Performance profile for all six Jacobi-type ADMM methods, the sixth method being (at least) twice as expensive per iteration as all other methods.

For the second performance profile in Figure 6, we also include the sixth method mentioned above. The comparison is again based on the iteration count, however, in this case one should take into account that each iteration of the sixth method, which needs to compute a certain step size at each iteration, is (at least) twice as expensive as all the other methods. Nevertheless, Figure 6 indicates that this step size rule makes this method more efficient, even more than Algorithm 3.1. On the other hand, even though Algorithm 3.1 works quite well, it was not our intention to create the fastest method, but to show that certain regularized Jacobi-type ADMM methods can be interpreted as a proximal-point method.

8 Final Remarks

We gave a proof of convergence for the regularized Jacobi-type ADMM method that is based on the proximal-point method or its modification by Halpern. This proximal-point interpretation allows some more freedom regarding the choice of certain parameters and allows itself a number of modifications. Even though we gave an example that shows that our proximal point interpretation may not converge for nonsymmetric matrices Q, the convergence of the regularized Gauss-Seidel alternating direction method for $N > 2$ remains an open question. The current technique of proof is not applicable in this setting, so that further investigations are necessary in this direction.

References

[1] H. Attouch and M. Soueycatt. Augmented Lagrangian and proximal alternating direction methods of multipliers in Hilbert spaces. Applications to games, PDE's and control. *Pacific Journal of Optimization*, 5(1):17–37, 2008.

[2] H. H. Bauschke and P. L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. CMS Books in Mathematics/Ouvrages de Mathématiques de la SMC. Springer, Cham, second edition, 2017.

[3] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific Optimization and Computation Series. Athena Scientific, Belmont, MA, third edition, 2016.

[4] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[5] H. Brezis and P. L. Lions. Produits infinis de resolvantes. *Israel Journal of Mathematics*, 29(4), 1978.

[6] C. Chen, B. He, Y. Ye, and X. Yuan. The direct extension of ADMM for multi-block convex minimization problems is not necessarily convergent. *Mathematical Programming*, 155(1-2, Ser. A):57–79, 2016.

[7] W. Deng, M.-J. Lai, Z. Peng, and W. Yin. Parallel multi-block ADMM with o $(1/k)$ convergence. *Journal of Scientific Computing*, 71(2):712–736, 2017.

[8] E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2):201–213, 2002.

[9] Y. Dong. The proximal point algorithm revisited. *Journal of Optimization Theory and Applications*, 161(2):478–489, 2014.

[10] Y. Dong. Comments on "The proximal point algorithm revisited". *Journal of Optimization Theory and Applications*, 166(1):343–349, 2015.

[11] J. Eckstein. Approximate iterations in Bregman-function-based proximal algorithms. *Mathematical Programming*, 83(1, Ser. A):113–123, 1998.

[12] J. Eckstein and W. Yao. Understanding the convergence of the alternating direction method of multipliers: Theoretical and computational perspectives. *Pacific Journal on Optimization*, 11(4):619– 644, 2015.

[13] F. Facchinei and J.-S. Pang. *Finite-Dimensional Variational Inequalities and Complementarity Problems*. Springer Series in Operations Research. Springer, 2003.

[14] R. Glowinski and P. Le Tallec. Augmented Lagrangian interpretation of the nonoverlapping Schwarz alternating method. In *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations (Houston, TX, 1989)*, number 43, pages 224 – 231. SIAM, Philadelphia, PA, 1990.

[15] G. Gu, B. He, and X. Yuan. Customized proximal point algorithms for linearly constrained convex minimization and saddle-point problems: a unified approach. *Computational Optimization and Applications*, 59(1-2):135, 2014.

[16] B. Halpern. Fixed points of nonexpanding maps. *Bulletin of the American Mathematical Society*, 73:957–961, 1967.

[17] B. He, L. Hou, and X. Yuan. On full Jacobian decomposition of the augmented Lagrangian method for separable convex programming. *SIAM Journal on Optimization*, 25(4):2274–2312, 2015.

[18] B. He, M. Tao, and X. Yuan. Alternating direction method with Gaussian back substitution for separable convex programming. *SIAM Journal on Optimization*, 22:313–340, 2012.

[19] B. He, M. Tao, and X. Yuan. Convergence rate analysis for the alternating direction method of multipliers with a substitution procedure for separable convex programming. *Mathematics of Operations Research*, 42:662–691, 2017.

[20] B. He, H.-K. Xu, and X. Yuan. On the proximal Jacobian decomposition of ALM for multiple-block separable convex minimization problems and its relationship to ADMM. *Journal of Scientific Computing*, 66(3):1204–1217, 2016.

[21] K. Ito and K. Kunisch. *Lagrange Multiplier Approach to Variational Problems and Applications*, volume 15 of *Advances in Design and Control*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2008.

[22] A. Iusem and R. G. Otero. Erratum: "Inexact versions of proximal point and augmented Lagrangian algorithms in Banach spaces" [Numer. Funct. Anal. Optim. **22** (2001), no. 5-6, 609–640; MR1849570 (2002e:90124)]. *Numerical Functional Analysis and Optimization. An International Journal*, 23(1-2):227–228, 2002.

[23] H. P. Langtangen and A. Logg. *Solving PDEs in Minutes–The FEniCS Tutorial Volume I*. Springer, 2017.

[24] S. Larsson and V. Thomée. *Partial Differential Equations with Numerical Methods*, volume 45. Springer Science & Business Media, 2008.

[25] P.-L. Lions. On the Schwarz alternating method. III. A variant for nonoverlapping subdomains. In *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations (Houston, TX, 1989)*, pages 202–223. SIAM, Philadelphia, PA, 1990.

[26] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer-Verlag, New York, 1999.

[27] R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

[28] R. T. Rockafellar. Monotone operators and the proximal point algorithm. *SIAM Journal on Control and Optimization*, 14(5):877–898, 1976.

[29] J. J. Wang and W. Song. An algorithm twisted from generalized ADMM for multi-block separable convex minimization models. *Journal of Computational and Applied Mathematics*, 309:342–358, 2017.

[30] X. Wang, M. Hong, S. Ma, and Z.-Q. Luo. Solving multiple-block separable convex minimization problems using two-block alternating direction method of multipliers. *arXiv preprint arXiv:1308.5294*, 2013.