universität wien

# MASTERARBEIT / MASTER'S THESIS

Titel der Masterarbeit / Title of the Master's Thesis

„Stochastic Primal-Dual Forward-Backward-Forward Algorithm with Applications in Machine Learning“


verfasst von / submitted by

Klaus Kastner


angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien, 2018 / Vienna 2018

| | |
|---|---|
| Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet: | A 066 821 |
| Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet: | Masterstudium Mathematik |
| Betreut von / Supervisor: | Univ.-Prof. Dr. Radu Ioan Boţ |

# Table of Contents



# 1 Introduction

There are many applications using primal-dual splitting methods in applied mathematics, e.g. evolution inclusions, partial differential equations, mechanics, variational inequalities, Nash equilibria, and various convex optimization problems such as support vector machine problems for classification and regression. The corresponding general problem which has to be solved is the monotone inclusion problem. In this thesis, we revisit the general primal-dual splitting framework presented in [2] from a stochastic point of view. We modify the therein provided algorithm by applying a random sweeping strategy, which selects the blocks of coordinates that are activated over the iterations. The sweeping rule allows for an arbitrary sampling of the indices of the coordinates. The resulting algorithm can be reduced to a stochastic error-tolerant version of Tseng's forward-backward-forward method in a product space. Essential for proving its almost sure convergence is the concept of quasi-Fejér monotonicity. Important properties of the stochastic primal-dual forward-backward-forward algorithm presented in this paper are that it tolerates errors in the implementation of the operators and that the operators involved are evaluated separately in each iteration, either by forward steps in the case of the single-valued ones, in particular the linear continuous operators and their adjoints, or by backward steps for set-valued ones, by using the corresponding resolvents. Furthermore, the algorithm is expected to converge faster than its deterministic counterpart, since every iteration computational effort is saved by using a sweeping strategy.

The problem under investigation is the following, we derive it by setting $z = 0$ and $r_i = 0$ for all $i \in \{1, ..., m\}$ in [2, Problem 1.1]. The parallel sum operator $\square$ is defined by Eq. 8.

### 1.1 Monotone Inclusion Problem

Let $\mathcal{H}$ be a separable real Hilbert space, let $m$ be a strictly positive integer, let $A : \mathcal{H} \to 2^{\mathcal{H}}$ be maximally monotone, and let $C : \mathcal{H} \to \mathcal{H}$ be monotone and $\mu$-Lipschitzian for some $\mu \in ]0,+\infty[$. For every $i \in \{1,...,m\}$, let $\mathcal{G}_i$ be a real separable Hilbert space, let $B_i : \mathcal{G}_i \to 2^{\mathcal{G}_i}$ be maximally monotone, let $D_i : \mathcal{G}_i \to 2^{\mathcal{G}_i}$ be monotone such that $D_i^{-1}$ is $\nu_i$-Lipschitzian, for some $\nu_i \in ]0,+\infty[$, and suppose that $L_i : \mathcal{H} \to \mathcal{G}_i$ is a nonzero bounded linear operator. The problem is to solve the primal inclusion

$$\text{(1)} \qquad \text{find} \quad \overline{x} \in \mathcal{H} \quad \text{such that} \quad 0 \in A\overline{x} + \sum_{i=1}^{m} L_i^* \big((B_i \,\square\, D_i)(L_i\overline{x})\big) + C\overline{x},$$

together with the dual inclusion

$$\text{(2)} \qquad \text{find } \overline{v}_1 \in \mathcal{G}_1, ..., \overline{v}_m \in \mathcal{G}_m \text{ such that } (\exists x \in \mathcal{H}) \begin{cases} -\sum_{i=1}^{m} L_i^* \overline{v}_i \in Ax + Cx \\ (\forall i \in \{1,...,m\}) \\ \quad \overline{v}_i \in (B_i \,\square\, D_i)(L_i x). \end{cases}$$

Problem 1.1 includes and extends diverse existing problem formulations, e.g. the convex optimization problem presented later on in Section 4.1.

In Section 2, we specify our notation and indicate the underlying stochastic assertion, which provides results about the convergence of quasi-Fejér monotone sequences. The main statement of this paper, which encompasses the stochastic primal-dual forward-backward-forward algorithm is formulated and proven in Section 3. As an offspring we obtain an algorithm in order to solve convex optimization problems. The corresponding theorem is presented and proven in Section 4.1. Moreover, in Section 4.2 we introduce the kernel based machine learning problem. In particular, we use a pool of hand-written images showing the numbers four or five in order to train a Support Vector Machine, which aims to classify the images correctly. This issue is equivalent to solving a convex minimization problem. By means of this application we test several types of the stochastic algorithm, assess their performances and compare them to the performance of its deterministic counterpart. At the end, the according MATLAB-codes are presented.

## 2 Background and Notation

We use the standard notation (cf. [1, Notation 2.1., Notation 3.1.], [2, 2 Notation and Background] and [8, 2 Preliminaries]). $\mathcal{H}$ is a separable real Hilbert space with scalar product $\langle \cdot, \cdot \rangle$, associated norm $\|\cdot\|$, and Borel $\sigma$-algebra $\mathcal{B}$. Id denotes the identity operator on $\mathcal{H}$ and $\rightharpoonup$ and $\to$ denote, respectively, weak and strong convergence in $\mathcal{H}$. The sets of weak and strong sequential cluster points of a sequence are denoted by $\mathcal{W}(x_n)_{n\geq 0}$ and $\mathcal{S}(x_n)_{n\geq 0}$, respectively. The underlying probability space is $(\Omega, \mathcal{F}, \mathrm{P})$. A $\mathcal{H}$-valued random variable is a measurable map $x : (\Omega, \mathcal{F}) \to (\mathcal{H}, \mathcal{B})$. The $\sigma$-algebra generated by a family $\Phi$ of random variables is denoted by $\sigma(\Phi)$. Let $\mathscr{F} = (\mathcal{F}_n)_{n\in\mathbb{N}}$ be a sequence of sub-sigma algebras of $\mathcal{F}$ such that $(\forall n \in \mathbb{N})\ \mathcal{F}_n \subset \mathcal{F}_{n+1}$. We denote by $\ell_+(\mathscr{F})$ the set of sequences of $[0,+\infty[$ -valued random variables $(\xi_n)_{n\in\mathbb{N}}$ such that, for every $n \in \mathbb{N}$, $\xi_n$ is $\mathcal{F}_n$-measurable. We set

$$\text{(3)} \qquad \big(\forall p \in ]0,+\infty[\big) \quad \ell_+^p(\mathscr{F}) = \Big\{ (\xi_n)_{n\in\mathbb{N}} \in \ell_+(\mathscr{F}) \,\Big|\, \sum_{n\in\mathbb{N}} \xi_n^p < +\infty \quad \text{P-a.s.} \Big\}.$$

Given a sequence $(x_n)_{n\in\mathbb{N}}$ of $\mathcal{H}$-valued random variables, we define

$$\mathscr{X} = (\mathcal{X}_n)_{n\in\mathbb{N}} \quad \text{where} \quad (\forall n \in \mathbb{N}) \quad \mathcal{X}_n = \sigma(x_0,...,x_n). \tag{4}$$

Equalities and inequalities involving random variables will always be understood to hold P-almost surely, even if the expression "P-a. s." is not explicitely written. For background on probability in Hilbert spaces, see [6] and [7].

For a strictly positive integer $m$ let $\mathcal{H}_1,...,\mathcal{H}_m$ be separable real Hilbert spaces and let $\mathcal{B}_1,...,\mathcal{B}_m$ be their corresponding Borel $\sigma$-algebras. Let $\mathcal{K} = \mathcal{H}_1 \oplus \cdots \oplus \mathcal{H}_m$ be their direct Hilbert sum with its corresponding Borel $\sigma$-algebra $\mathcal{B}$, which is equal to the product $\sigma$-algebra $\bigotimes_{i=1,...,m} \mathcal{B}_i$, i.e. a function from $\Omega$ to $\mathcal{K}$ is $\mathcal{B}$-measurable if and only if its $i$-th component is $\mathcal{B}_i$-measurable for $i = 1,...,m$. The scalar products and associated norms of these spaces are all denoted by $\langle\cdot,\cdot\rangle$ and $||\cdot||$, respectively, and $\mathbf{x} = (x_1,...,x_m)$ denotes a generic vector in in $\mathcal{K}$, which remains a real separable Hilbert space with respect to the inner product $\langle \mathbf{x},\mathbf{y}\rangle = \sum_{i=1}^m \langle x_i, y_i\rangle$ and the norm $||\mathbf{x}|| = \sqrt{\sum_{i=1}^m ||x_i||^2}$. Given a sequence $(\mathbf{x}_n)_{n\in\mathbb{N}} = (x_{1,n},...,x_{m,n})_{n\in\mathbb{N}}$ of $\mathcal{K}$-valued random variables, we set $(\forall n \in \mathbb{N})$ $\mathbf{\mathcal{X}}_n = \sigma(\mathbf{x}_0,...,\mathbf{x}_n)$.

Let $M : \mathcal{H} \to 2^{\mathcal{H}}$ be a set-valued operator. We denote by $\operatorname{ran} M = \{u \in \mathcal{H} \mid (\exists x \in \mathcal{H})\ u \in Mx\}$ the range of $M$, by $\operatorname{dom} M = \{x \in \mathcal{H} \mid Mx \neq \emptyset\}$ its domain, by $\operatorname{zer} M = \{x \in \mathcal{H} \mid 0 \in Mx\}$ its set of zeros, by $\operatorname{gra} M = \{(x,u) \in \mathcal{H}\times\mathcal{H} \mid u \in Mx\}$ its graph, and by $M^{-1}$ its inverse, i.e., the set-valued operator with graph $\{(u,x) \in \mathcal{H}\times\mathcal{H} \mid u \in Mx\}$. The resolvent of $M$ is

$$J_M = (\mathrm{Id} + M)^{-1}. \tag{5}$$

Moreover, $M$ is monotone if

$$(\forall (x,u) \in \operatorname{gra} M)(\forall (y,v) \in \operatorname{gra} M) \quad \langle x-y, u-v\rangle \geq 0, \tag{6}$$

and maximally monotone if there exists no monotone operator $\tilde{M} : \mathcal{H} \to 2^{\mathcal{H}}$ such that $\operatorname{gra} M \subsetneq \operatorname{gra} \tilde{M}$. We say that $M$ is uniformly monotone at $x \in \operatorname{dom} M$ if there exists an increasing function $\phi : [0,+\infty[ \to [0,+\infty]$ that vanishes only at 0 such that

$$(\forall u \in Mx)(\forall (y,v) \in \operatorname{gra} M) \quad \langle x-y, u-v\rangle \geq \phi(||x-y||). \tag{7}$$

The parallel sum of two set-valued operators $M_1$ and $M_2$ from $\mathcal{H}$ to $2^{\mathcal{H}}$ is

$$M_1 \ \square\ M_2 = (M_1^{-1} + M_2^{-1})^{-1}. \tag{8}$$

This operation is very important in convex analysis and monotone operator theory, and its is naturally connected to addition since $(M_1+M_2)^{-1} = M_1^{-1} \ \square\ M_2^{-1}$. One can find more information about the parallel sum in [3]. Let be $\mathcal{H}_1$ and $\mathcal{H}_2$ two separable real Hilbert spaces with corresponding scalar products $\langle\cdot,\cdot\rangle_1$ and $\langle\cdot,\cdot\rangle_2$, respectively. For a linear continuous operator $L : \mathcal{H}_1 \to \mathcal{H}_2$, the operator $L^* : \mathcal{H}_2 \to \mathcal{H}_1$, defined via $\langle Lx, y\rangle_2 = \langle x, L^*y\rangle_1$ for all $x \in \mathcal{H}_1$ and all $y \in \mathcal{H}_2$, denotes its adjoint. We denote by $\Gamma_0(\mathcal{H})$ the class of lower semicontinuous convex functions $\varphi : \mathcal{H} \to ]-\infty,+\infty]$ such that $\operatorname{dom}\varphi = \{x \in \mathcal{H} \mid \phi(x) < +\infty\} \neq \emptyset$. Now let $\varphi \in \Gamma_0(\mathcal{H})$. The conjugate of $\varphi$ is the function $\varphi^* \in \Gamma_0(\mathcal{H})$ defined by $\varphi^* : u \mapsto \sup_{x\in\mathcal{H}}(\langle x,u\rangle - \phi(x))$, and the subdifferential of $\varphi$ is the maximally monotone operator

$$\partial\varphi : \mathcal{H} \to 2^{\mathcal{H}} : x \mapsto \{u \in \mathcal{H} \mid (\forall y \in \mathcal{H})\ \langle y-x, u\rangle + \varphi(x) \leq \varphi(y)\} \tag{9}$$

with inverse given by

$$(\partial\varphi)^{-1} = \partial\varphi^*. \tag{10}$$

Moreover, for every $x \in \mathcal{H}$, $\varphi + ||x - \cdot||^2/2$ possesses a unique minimizer, which is denoted by $\operatorname{prox}_\varphi x$. We have

$$\operatorname{prox}_\varphi = J_{\partial\varphi}. \tag{11}$$

We say that $\varphi$ is $\nu$-strongly convex for some $\nu \in ]0,+\infty[$ if $\varphi - \nu||\cdot||^2/2$ is convex, and that $\varphi$ is uniformly convex at $x \in \operatorname{dom}\varphi$ if there exists an increasing function $\phi : [0,+\infty[ \to [0,+\infty]$ that vanishes only at 0 such that

$$\begin{aligned}(\forall y \in \operatorname{dom}\varphi)(\forall \alpha \in ]0,1[)\ \ &\varphi(\alpha x + (1-\alpha)y) + \alpha(1-\alpha)\phi(||x-y||)\\ &\leq \alpha\varphi(x) + (1-\alpha)\varphi(y).\end{aligned} \tag{12}$$

The infimal convolution of two functions $\varphi_1$ and $\varphi_2$ from $\mathcal{H}$ to $]-\infty,+\infty]$ is

$$\varphi_1 \,\square\, \varphi_2 : \mathcal{H} \to [-\infty,+\infty] : x \mapsto \inf_{y\in\mathcal{H}} (\varphi_1(y) + \varphi_2(x-y)). \tag{13}$$

Finally, let $S$ be a convex subset of $\mathcal{H}$. The strong relative interior of $S$, i.e., the set of points $x$ in $S$ such that the cone generated by $-x+S$ is a closed vector subspace of $\mathcal{H}$, is denoted by $\operatorname{sri} S$, and the relative interior in $S$, i.e., the set of points $x$ in $S$ such that the cone generated by $-x+S$ is a vector subspace of $\mathcal{H}$, is denoted by $\operatorname{ri} S$.

## 2.1 Stochastic Foundation

The following result is a main ingredient in the prove of the stochastic primal-dual forward-backward-forward algorithm. A sequence satisfying Eq. 14 P-a. s. is referred to as *quasi-Fejér monotone*.

**Theorem 1 (see [1, PROPOSITION 2.3.])** *Let $F$ be a nonempty closed subset of $\mathcal{H}$, let $\phi : [0,+\infty[ \to [0,+\infty[$ be a strictly increasing function such that $\lim_{t\to+\infty}\phi(t) = +\infty$, and let $(x_n)_{n\in\mathbb{N}}$ be a sequence of $\mathcal{H}$-valued random variables. Suppose that, for every $z \in F$, there exist $(\chi_n(z)_{n\in\mathbb{N}}) \in \ell^1_+(\mathcal{X})$, $(\vartheta_n(z)_{n\in\mathbb{N}}) \in \ell_+(\mathcal{X})$, and $(\eta_n(z)_{n\in\mathbb{N}}) \in \ell^1_+(\mathcal{X})$ such that the following is satisfied* P-a. s.*:*

$$(\forall n \in \mathbb{N})\ \ \mathrm{E}(\phi(||x_{n+1}-z||)\,|\,\mathcal{X}_n) + \vartheta_n(z) \leq (1+\chi_n(z))\phi(||x_n-z||) + \eta_n(z) \tag{14}$$

*Then the following hold:*

(i) $(\forall z \in \mathcal{F})\ \left[\sum_{n\in\mathbb{N}} \vartheta_n(z) < +\infty \ \text{ P-a. s.}\right]$.

(ii) $(x_n)_{n\in\mathbb{N}}$ *is bounded* P-a. s.

(iii) *There exists $\tilde{\Omega} \in \mathcal{F}$ such that $\mathrm{P}(\tilde{\Omega}) = 1$ and, for every $\omega \in \tilde{\Omega}$ and every $z \in F$, $(||x_n(\omega) - z||)_{n\in\mathbb{N}}$ converges.*

(iv) *Suppose that $\mathcal{W}(x_n)_{n\in\mathbb{N}} \subset F$* P-a. s. *Then $(x_n)_{n\in\mathbb{N}}$ converges weakly* P-a. s. *to an $F$-valued random variable.*

(v) *Suppose that $\mathcal{S}(x_n)_{n\in\mathbb{N}} \cap F \neq \emptyset$* P-a. s. *Then $(x_n)_{n\in\mathbb{N}}$ converges strongly* P-a. s. *to an $F$-valued random variable.*

(vi) *Suppose that $\mathcal{S}(x_n)_{n\in\mathbb{N}} \neq \emptyset$* P-a. s. *and that $\mathcal{W}(x_n)_{n\in\mathbb{N}} \subset F$* P-a. s. *Then $(x_n)_{n\in\mathbb{N}}$ converges strongly* P-a. s. *to an $F$-valued random variable.*

# 3 Main Result

The following statement is the main result of this thesis, it presents the new stochastic primal-dual forward-backward-forward algorithm and describes its asymptotic behavior.

**Theorem 2 (Main Result)** *In the Problem given in Subsection 1.1, suppose that*

$$0 \in \operatorname{ran}\Big(A + \sum_{i=1}^{m} L_i^*\big((B_i \,\square\, D_i)(L_i\,\cdot)\big) + C\Big). \tag{15}$$

*Let $(a_{1,n})_{n\in\mathbb{N}}$, $(b_{1,n})_{n\in\mathbb{N}}$ and $(c_{1,n})_{n\in\mathbb{N}}$ be sequences of $\mathcal{H}$-valued random variables, and for every $i \in \{1,...,m\}$, let $(a_{2,i,n})_{n\in\mathbb{N}}$, $(b_{2,i,n})_{n\in\mathbb{N}}$ and $(c_{2,i,n})_{n\in\mathbb{N}}$ be sequences of $\mathcal{G}_i$-valued random variables. Furthermore, set*

$$\beta = \max\{\mu, \nu_1, ..., \nu_m\} + \sqrt{\sum_{i=1}^{m} ||L_i||^2}, \tag{16}$$

*let $x_0$ be a $\mathcal{H}$-valued random variable, let $(v_{1,0}, ..., v_{m,0})$ be a $\mathcal{G}_1 \oplus \cdots \oplus \mathcal{G}_m$-valued vector of random variables, let $\rho \in \left]0, 1/(\beta+1)\right[$, let $(\gamma_n)_{n\in\mathbb{N}}$ be a sequence in $[\rho, (1-\rho)/\beta]$. Set $\mathbf{D} = \{0,1\}^{m+1} \setminus \{\mathbf{0}\}$, and let for all $n \in \mathbb{N}$ $(\varepsilon_{1,n}, \varepsilon_{2,1,n}, ..., \varepsilon_{2,m,n})$ be identically distributed $\mathbf{D}$-valued random variables, and iterate*

**Algorithm 1:**

**for** $n = 0, 1, ...$ **do**
  $y_{1,n} = x_n - \gamma_n(Cx_n + \sum_{i=1}^{m} L_i^* v_{i,n} + a_{1,n})$
  $p_{1,n} = J_{\gamma_n A}(y_{1,n}) + b_{1,n}$
  **if** $\varepsilon_{1,n} = 0$ **then**
    **for** $i = 1, ..., m$ **do**
      **if** $\varepsilon_{2,i,n} = 1$ **then**
        $y_{2,i,n} = v_{i,n} + \gamma_n(L_i x_n - D_i^{-1} v_{i,n} - a_{2,i,n})$
        $p_{2,i,n} = J_{\gamma_n B_i^{-1}}(y_{2,i,n}) + b_{2,i,n}$
        $q_{2,i,n} = p_{2,i,n} + \gamma_n(L_i p_{1,n} - D_i^{-1} p_{2,i,n} - c_{2,i,n})$
        $v_{i,n+1} = v_{i,n} - y_{2,i,n} + q_{2,i,n}$
      **else**
        $v_{i,n+1} = v_{i,n}$
    $x_{n+1} = x_n$
  **else** $[\varepsilon_{1,n} = 1]$
    **for** $i = 1, ..., m$ **do**
      $y_{2,i,n} = v_{i,n} + \gamma_n(L_i x_n - D_i^{-1} v_{i,n} - a_{2,i,n})$
      $p_{2,i,n} = J_{\gamma_n B_i^{-1}}(y_{2,i,n}) + b_{2,i,n}$
    $q_{1,n} = p_{1,n} - \gamma_n(Cp_{1,n} + \sum_{i=1}^{m} L_i^* p_{2,i,n} + c_{1,n})$
    $x_{n+1} = x_n - y_{1,n} + q_{1,n}$
    **for** $i = 1, ..., m$ **do**
      **if** $\varepsilon_{2,i,n} = 1$ **then**
        $q_{2,i,n} = p_{2,i,n} + \gamma_n(L_i p_{1,n} - D_i^{-1} p_{2,i,n} - c_{2,i,n})$
        $v_{i,n+1} = v_{i,n} - y_{2,i,n} + q_{2,i,n}$
      **else**
        $v_{i,n+1} = v_{i,n}$

*and for all* $n \in \mathbb{N}$ *set* $\mathscr{E}_n = \sigma\big((\varepsilon_{1,n}, \varepsilon_{2,1,n}, ..., \varepsilon_{2,m,n})\big)$, $\mathcal{X}_{1,n} = \sigma(x_0, ..., x_n)$ *and* $(\forall i \in \{1, ..., m\})\ \mathcal{X}_{2,i,n} = \sigma(v_{2,i,0}, ..., v_{2,i,n})$. *In addition, assume that the following hold.*

(A) *For all* $n \in \mathbb{N}$, *let* $a_{1,n}$, $b_{1,n}$ *and* $c_{1,n}$ *be* $\mathcal{X}_{1,n}$*-measurable and* $(\forall i \in \{1, ..., m\})$ *let* $a_{2,i,n}$, $b_{2,i,n}$ *and* $c_{2,i,n}$ *be* $\mathcal{X}_{2,i,n}$*-measurable.*
*Further, let* $\sum_{n\in\mathbb{N}} \mathrm{E}(||a_{1,n}|| \,|\, \mathcal{X}_{1,n}) < +\infty$, $\sum_{n\in\mathbb{N}} \mathrm{E}(||b_{1,n}|| \,|\, \mathcal{X}_{1,n}) < +\infty$, $\sum_{n\in\mathbb{N}} \mathrm{E}(||c_{1,n}|| \,|\, \mathcal{X}_{1,n}) < +\infty$, *and let for all* $i \in \{1, ..., m\}$ $\sum_{n\in\mathbb{N}} \mathrm{E}(||a_{2,i,n}|| \,|\, \mathcal{X}_{2,i,n}) < +\infty$, $\sum_{n\in\mathbb{N}} \mathrm{E}(||b_{2,i,n}|| \,|\, \mathcal{X}_{2,i,n}) < +\infty$ *and* $\sum_{n\in\mathbb{N}} \mathrm{E}(||c_{2,i,n}|| \,|\, \mathcal{X}_{2,i,n}) < +\infty$.

(B) *For every* $n \in \mathbb{N}$, *let* $\mathscr{E}_n$ *and* $\sigma(\mathcal{X}_{1,n}, \mathcal{X}_{2,1,n}, ..., \mathcal{X}_{2,m,n})$ *be independent.*

(C) *Let* $\mathrm{P}[\varepsilon_{1,0} = 1] > 0$ *and for all* $i \in \{1, ..., m\}$ *let* $\mathrm{P}[\varepsilon_{2,i,0} = 1] > 0$.

*Then the following hold.*

(i) $\sum_{n\in\mathbb{N}} ||x_n - p_{1,n}||^2 < +\infty$ P-a. s. *and* $(\forall i \in \{1, ..., m\}) \sum_{n\in\mathbb{N}} ||v_{i,n} - p_{2,i,n}||^2 < +\infty$ P-a. s.

(ii) *There exists a solution* $\overline{x}$ *to Eq. 1 and a solution* $(\overline{v}_1, ..., \overline{v}_m)$ *to Eq. 2 such that the following* P-a. s. *hold.*

(a) $-\sum_{j=1}^m L_j^* \overline{v}_j \in A\overline{x} + C\overline{x}$ *and* $(\forall i \in \{1, ..., m\})\ L_i\overline{x} \in B_i^{-1}\overline{v}_i + D_i^{-1}\overline{v}_i$.

(b) $(\forall i \in \{1, ..., m\})\ 0 \in -L_i\big((A^{-1} \,\square\, C^{-1})(-\sum_{j=1}^m L_j^* \overline{v}_j)\big) + B_i^{-1}\overline{v}_i + D_i^{-1}\overline{v}_i$.

(c) $x_n \rightharpoonup \overline{x}$ *and* $p_{1,n} \rightharpoonup \overline{x}$.

(d) $(\forall i \in \{1, ..., m\})\ v_{i,n} \rightharpoonup \overline{v}_i$ *and* $p_{2,i,n} \rightharpoonup \overline{v}_i$.

(e) *Suppose that* $A$ *or* $C$ *is uniformly monotone at* $\overline{x}$. *Then* $x_n \to \overline{x}$ *and* $p_{1,n} \to \overline{x}$.

(f) *Suppose that, for some* $i \in \{1, ..., m\}$, $B_i^{-1}$ *or* $D_i^{-1}$ *is uniformly monotone at* $\overline{v}_i$. *Then* $v_{i,n} \to \overline{v}_i$ *and* $p_{2,i,n} \to \overline{v}_i$.

In order to prove the above result, we use the following Theorem 3. Therein a stochastic error-tolerant forward-backward-forward algorithm and its corresponding convergence behavior is formulated. The concept of the proof of Theorem 2 (Main Result) is basically to transform Algorithm 1 into the algorithm given in the following theorem and then apply it.

**Theorem 3 (A stochastic error-tolerant forward-backward-forward Alg.)** *For a strictly positive integer* $m$ *let* $\mathcal{H}_1, ..., \mathcal{H}_m$ *be separable real Hilbert spaces and* $\mathcal{K} = \mathcal{H}_1 \oplus \cdots \oplus \mathcal{H}_m$ *be their direct Hilbert sum with corresponding Borel* $\sigma$*-algebra* $\mathcal{B}$. *Let for all* $i$ *in* $\{1, ..., m\}$ $M_i : \mathcal{K} \to 2^{\mathcal{H}_i} : \mathbf{x} = (x_1, ..., x_m) \mapsto M_i x_i$ *and suppose that* $\mathbf{M} : \mathcal{K} \to 2^{\mathcal{K}} : \mathbf{x} \mapsto \prod_{i=1}^m M_i x_i$ *is maximally monotone. Moreover, let for all* $i$ *in* $\{1, ..., m\}$ $Q_i : \mathcal{K} \to \mathcal{H}_i : \mathbf{x} \mapsto Q_i\mathbf{x}$ *be single-valued and let* $\mathbf{Q} : \mathcal{K} \to \mathcal{K} : \mathbf{x} \mapsto \prod_{i=1}^m Q_i\mathbf{x}$ *be a monotone and* $\beta$*-Lipschitz continuous operator, for some* $\beta > 0$. *Suppose that* $\mathrm{zer}(\mathbf{M} + \mathbf{Q}) \neq \emptyset$. *Let* $\mathbf{x}_0$ *be a* $\mathcal{K}$*-valued random variable, let* $(\mathbf{a}_n)_{n\in\mathbb{N}}$, $(\mathbf{b}_n)_{n\in\mathbb{N}}$ *and* $(\mathbf{c}_n)_{n\in\mathbb{N}}$ *be sequences of* $\mathcal{K}$*-valued random variables, let* $\rho \in \left]0, 1/(1-\beta)\right[$, *and let* $(\gamma_n)_{n\in\mathbb{N}}$ *be a sequence*

*in* $[\rho,(1-\rho)/\beta]$. *Set* $\mathbf{D}=\{0,1\}^m\setminus\{\mathbf{0}\}$ *and let* $(\boldsymbol{\varepsilon}_n=(\varepsilon_{1,n},...,\varepsilon_{m,n}))_{n\in\mathbb{N}}$ *be identically distributed* $\mathbf{D}$*-valued random variables. Iterate*

**Algorithm 2:**

1 **for** $n=0,1,...$ **do**
2 $\quad\mathbf{y}_n=\mathbf{x}_n-\gamma_n(\mathbf{Q}\mathbf{x}_n+\mathbf{a}_n)$
3 $\quad\mathbf{z}_n=\mathbf{J}_{\gamma_n\mathbf{M}}(\mathbf{y}_n)+\mathbf{b}_n$
4 $\quad\mathbf{q}_n=\mathbf{z}_n-\gamma_n(\mathbf{Q}\mathbf{z}_n+\mathbf{c}_n)$
5 $\quad$**for** $i=1,...,m$ **do**
6 $\quad\quad x_{i,n+1}=x_{i,n}+\varepsilon_{i,n}(q_{i,n}-y_{i,n})$
7 $\quad\mathbf{x}_{n+1}=(x_{1,n+1},...,x_{m,n+1})$

*and for all* $n\in\mathbb{N}$ *set* $\mathscr{E}_n=\sigma(\boldsymbol{\varepsilon}_n)$, *and* $\mathfrak{X}_n=\sigma(\mathbf{x}_1,...,\mathbf{x}_n)$ *und set* $\mathscr{X}=(\mathfrak{X}_n)_{n\in\mathbb{N}}$. *In addition, assume that the following hold.*

(i) *For every* $n\in\mathbb{N}$, *let* $\mathbf{a}_n$, $\mathbf{b}_n$ *and* $\mathbf{c}_n$ *be* $\mathfrak{X}_n$*-measurable, and let* $\sum_{n\in\mathbb{N}}\mathrm{E}(||\mathbf{a}_n||\,|\,\mathfrak{X}_n)<+\infty$, $\sum_{n\in\mathbb{N}}\mathrm{E}(||\mathbf{b}_n||\,|\,\mathfrak{X}_n)<+\infty$ *and* $\sum_{n\in\mathbb{N}}\mathrm{E}(||\mathbf{c}_n||\,|\,\mathfrak{X}_n)<+\infty$.

(ii) *For every* $n\in\mathbb{N}$, *let* $\mathscr{E}_n$ *and* $\mathfrak{X}_n$ *be independent.*

(iii) $(\forall i\in\{1,...,m\})\;\; p_i=\mathrm{P}[\varepsilon_{i,0}=1]>0$.

*Then the following hold for some random variable* $\overline{\mathbf{x}}\in\mathrm{zer}(\mathbf{M}+\mathbf{Q})$.

(a) $\sum_{n\in\mathbb{N}}||\mathbf{x}_n-\mathbf{z}_n||^2<+\infty$ P-a. s.

(b) $\mathbf{x}_n\rightharpoonup\overline{\mathbf{x}}$ P-a. s. *and* $\mathbf{z}_n\rightharpoonup\overline{\mathbf{x}}$ P-a. s.

(c) *Suppose that* $\mathbf{M}$ *or* $\mathbf{Q}$ *is uniformly monotone at* $\overline{\mathbf{x}}$. *Then* $\mathbf{x}_n\to\overline{\mathbf{x}}$ P-a. s. *and* $\mathbf{z}_n\to\overline{\mathbf{x}}$ P-a. s.

*Proof to Theorem 3.* (The paradigms to this proof are the proofs of [1, THEOREM 3.2], [3, Theorem 25.10-Tsengs algorithm] and [5, Theorem 2.5].) First we notice that Algorithm 2 is well defined, since for all $n\in\mathbb{N}$ $\mathbf{J}_{\gamma_n\mathbf{M}}$ is single-valued [3, Corollary 23.8]. We define the linear operators

$$(\forall n\in\mathbb{N})\;\; \mathbf{T}_{\boldsymbol{\varepsilon}_n}:\mathcal{K}\to\mathcal{K}:(x_1,...,x_m)\mapsto(\varepsilon_{1,n}x_1,...,\varepsilon_{m,n}x_m), \tag{17}$$

and set

$$(\forall n\in\mathbb{N})\begin{cases}\tilde{\mathbf{y}}_n=\mathbf{x}_n-\gamma_n\mathbf{Q}\mathbf{x}_n & \tilde{\mathbf{y}}_n=(\tilde{y}_{1,n},...,\tilde{y}_{m,n})\\ \tilde{\mathbf{z}}_n=\mathbf{J}_{\gamma_n\mathbf{M}}(\tilde{\mathbf{y}}_n) & \tilde{\mathbf{z}}_n=(\tilde{z}_{1,n},...,\tilde{z}_{m,n})\\ \tilde{\mathbf{q}}_n=\tilde{\mathbf{z}}_n-\gamma_n\mathbf{Q}\tilde{\mathbf{z}}_n & \tilde{\mathbf{q}}_n=(\tilde{q}_{1,n},...,\tilde{q}_{m,n})\\ \mathbf{e}_n=\mathbf{y}_n-\mathbf{q}_n-\tilde{\mathbf{y}}_n+\tilde{\mathbf{q}}_n & \mathbf{e}_n=(e_{1,n},...,e_{m,n})\\ \tilde{\mathbf{e}}_n=\mathbf{T}_{\boldsymbol{\varepsilon}_n}(\mathbf{e}_n) & \tilde{\mathbf{e}}_n=(\tilde{e}_{1,n},...,\tilde{e}_{m,n}).\end{cases} \tag{18}$$

Thus the lines 5 to 7 in Algorithm 2 read as

$$\mathbf{x}_{n+1}=\mathbf{x}_n+\mathbf{T}_{\boldsymbol{\varepsilon}_n}(\mathbf{q}_n-\mathbf{y}_n). \tag{19}$$

We define a norm $|||\cdot|||$ on $\mathcal{K}$ by

$$(\forall\mathbf{x}\in\mathcal{K})\;\; |||\mathbf{x}|||^2=\sum_{i=1}^m\frac{1}{p_i}||x_i||^2, \tag{20}$$

where $p_i = \mathrm{P}[\varepsilon_{i,0} = 1] > 0$ as defined in (iii). We are going to apply Theorem 1 in $(\mathcal{K}, |||\cdot|||)$. For $\mathbf{z} \in \mathrm{zer}(\mathbf{M}+\mathbf{Q})$, let us set

$$
\begin{aligned}
&(\forall n \in \mathbb{N})(\forall i \in \{1,...,m\}) \\
&\quad \tilde{h}_{i,n} : \mathcal{K} \times \mathbf{D} \to \mathbb{R} : (\mathbf{x}_n, \boldsymbol{\varepsilon}_n) \mapsto ||x_{i,n} + \varepsilon_{i,n}(\tilde{q}_{i,n} - \tilde{y}_{i,n}) - z_i||^2 \text{ and} \\
&\quad h_{i,n} : \mathcal{K}^4 \times \mathbf{D} \to \mathbb{R} : (\mathbf{x}_n, \mathbf{a}_n, \mathbf{b}_n, \mathbf{c}_n, \boldsymbol{\varepsilon}_n) \mapsto ||x_{i,n+1} - z_i||^2 \\
&\qquad\qquad\qquad\qquad\qquad\qquad = ||x_{i,n} + \varepsilon_{i,n}(q_{i,n} - y_{i,n}) - z_i||^2.
\end{aligned} \tag{21}
$$

Since $\mathbf{M}$ is maximally monotone, $\mathbf{J}_{\gamma_n \mathbf{M}}$ is firmly nonexpansive [3, Corollary 23.8], hence continuous. Note that, for every $n \in \mathbb{N}$ and every $i \in \{1,...,m\}$, since $\mathbf{Q}$ and $\mathbf{J}_{\gamma_n \mathbf{M}}$ are continuous and hence measurable, so are the functions $(\tilde{h}_{i,n}(\cdot, \boldsymbol{\eta}))_{\boldsymbol{\eta} \in \mathbf{D}}$ $\mathcal{B}$-measurable and $(h_{i,n}(\cdot,\cdot,\cdot,\cdot,\boldsymbol{\eta}))_{\boldsymbol{\eta} \in \mathbf{D}}$ $\bigotimes_{i=1}^4 \mathcal{B}$-measurable. Consequently, since, for every $n \in \mathbb{N}$, (ii) asserts that the events $([\boldsymbol{\varepsilon}_n = \boldsymbol{\eta}])_{\boldsymbol{\eta} \in \mathbf{D}}$ form an almost sure partition of $\Omega$ and are independent from $\mathfrak{X}_n$, and since the random variables $(\tilde{h}_{i,n}(\mathbf{x}_n, \boldsymbol{\eta}))_{1 \le i \le m, \boldsymbol{\eta} \in \mathbf{D}}$ and
$(h_{i,n}(\mathbf{x}_n, \mathbf{a}_n, \mathbf{b}_n, \mathbf{c}_n, \boldsymbol{\eta}))_{1 \le i \le m, \boldsymbol{\eta} \in \mathbf{D}}$ are $\mathfrak{X}_n$-measurable, we obtain [4, Section 28.2]

$$
\begin{aligned}
(\forall n \in \mathbb{N})(\forall i \in \{1,...,m\}) \ \mathrm{E}(\tilde{h}_{i,n}(\mathbf{x}_n, \boldsymbol{\varepsilon}_n) \,|\, \mathfrak{X}_n) &= \mathrm{E}(\tilde{h}_{i,n}(\mathbf{x}_n, \boldsymbol{\varepsilon}_n) \sum_{\boldsymbol{\eta} \in \mathbf{D}} 1_{\{\boldsymbol{\varepsilon}_n = \boldsymbol{\eta}\}} \,|\, \mathfrak{X}_n) \\
&= \sum_{\boldsymbol{\eta} \in \mathbf{D}} \mathrm{E}(\tilde{h}_{i,n}(\mathbf{x}_n, \boldsymbol{\eta}) 1_{\{\boldsymbol{\varepsilon}_n = \boldsymbol{\eta}\}} \,|\, \mathfrak{X}_n) \\
&= \sum_{\boldsymbol{\eta} \in \mathbf{D}} \mathrm{E}(1_{\{\boldsymbol{\varepsilon}_n = \boldsymbol{\eta}\}} \,|\, \mathfrak{X}_n) \tilde{h}_{i,n}(\mathbf{x}_n, \boldsymbol{\eta}) \\
&= \sum_{\boldsymbol{\eta} \in \mathbf{D}} \mathrm{P}[\boldsymbol{\varepsilon}_n = \boldsymbol{\eta}] \tilde{h}_{i,n}(\mathbf{x}_n, \boldsymbol{\eta}) \ \text{ P-a.s.},
\end{aligned} \tag{22}
$$

and analogously,

$$
\begin{aligned}
&(\forall n \in \mathbb{N})(\forall i \in \{1,...,m\}) \ \mathrm{E}(h_{i,n}(\mathbf{x}_n, \mathbf{a}_n, \mathbf{b}_n, \mathbf{c}_n, \boldsymbol{\varepsilon}_n) \,|\, \mathfrak{X}_n) \\
&\qquad\qquad = \sum_{\boldsymbol{\eta} \in \mathbf{D}} \mathrm{P}[\boldsymbol{\varepsilon}_n = \boldsymbol{\eta}] h_{i,n}(\mathbf{x}_n, \mathbf{a}_n, \mathbf{b}_n, \mathbf{c}_n, \boldsymbol{\eta}) \ \text{ P-a.s.}
\end{aligned} \tag{23}
$$

Since $\mathbf{Q} : \mathcal{K} \to \mathcal{K}$ is monotone and continuous, it is maximally monotone [3, Example 20.29]. Furthermore, since $\mathrm{dom}\, \mathbf{Q} = \mathcal{K}$, we derive from [3, Corollary 24.4(i)] that

$$
\mathbf{M} + \mathbf{Q} \text{ is maximally monotone.} \tag{24}
$$

Since, for all $n \in \mathbb{N}$, $\mathbf{a}_n$ is $\mathfrak{X}_n$-measurable, we have $\mathrm{E}(||\mathbf{a}_n|| \,|\, \mathfrak{X}_n) = ||\mathbf{a}_n||$ P-a.s., i.e., for all $n \in \mathbb{N}$ exists a $\Omega_n$ with $\mathrm{P}(\Omega_n) = 1$ such that for all $\omega \in \Omega_n$ $\mathrm{E}(||\mathbf{a}_n(\omega)|| \,|\, \mathfrak{X}_n) = ||\mathbf{a}_n(\omega)||$. Now set $\tilde{\Omega} = \bigcap_{n \in \mathbb{N}} \Omega_n$ and let $\tilde{\Omega}^c$ be its complement, then $\mathrm{P}(\tilde{\Omega}) = 1 - \mathrm{P}(\bigcup_{n \in \mathbb{N}} \Omega_n^c) \ge 1 - \sum_{n \in \mathbb{N}} \mathrm{P}(\Omega_n^c) = 1$, hence $\mathrm{P}(\tilde{\Omega}) = 1$. We have for all $\omega \in \tilde{\Omega}$ and for all $n \in \mathbb{N}$ that $\mathrm{E}(||\mathbf{a}_n(\omega)|| \,|\, \mathfrak{X}_n) = ||\mathbf{a}_n(\omega)||$. Analogously, we get the same result for $\mathbf{b}_n$ and $\mathbf{c}_n$, respectively, instead of $\mathbf{a}_n$. Together with (i) we derive

$$
\begin{aligned}
&\sum_{n \in \mathbb{N}} ||\mathbf{a}_n|| < +\infty \ \text{ P-a.s.}, \sum_{n \in \mathbb{N}} ||\mathbf{b}_n|| < +\infty \ \text{ P-a.s. and} \\
&\sum_{n \in \mathbb{N}} ||\mathbf{c}_n|| < +\infty \ \text{ P-a.s.}
\end{aligned} \tag{25}
$$

We derive from Algorithm 2 and Eq. 18 the following inequalities. First,

$$||\tilde{\mathbf{y}}_n - \mathbf{y}_n|| = \gamma_n ||\mathbf{a}_n|| \leq ||\mathbf{a}_n||/\beta. \tag{26}$$

Hence, since $\mathbf{J}_{\gamma_n \mathbf{M}}$ is nonexpansive,

$$\begin{aligned} ||\tilde{\mathbf{z}}_n - \mathbf{z}_n|| &= ||\mathbf{J}_{\gamma_n \mathbf{M}}(\tilde{\mathbf{y}}_n) - \mathbf{J}_{\gamma_n \mathbf{M}}(\mathbf{y}_n) - \mathbf{b}_n|| \\ &\leq ||\mathbf{J}_{\gamma_n \mathbf{M}}(\tilde{\mathbf{y}}_n) - \mathbf{J}_{\gamma_n \mathbf{M}}(\mathbf{y}_n)|| + ||\mathbf{b}_n|| \\ &\leq ||\tilde{\mathbf{y}}_n - \mathbf{y}_n|| + ||\mathbf{b}_n|| \\ &\leq ||\mathbf{a}_n||/\beta + ||\mathbf{b}_n||. \end{aligned} \tag{27}$$

In turn, we get

$$\begin{aligned} ||\tilde{\mathbf{q}}_n - \mathbf{q}_n|| &= ||\tilde{\mathbf{z}}_n - \gamma_n \mathbf{Q}\tilde{\mathbf{z}}_n - \mathbf{z}_n + \gamma_n(\mathbf{Q}\mathbf{z}_n + \mathbf{c}_n)|| \\ &\leq ||\tilde{\mathbf{z}}_n - \mathbf{z}_n|| + \gamma_n ||\mathbf{Q}\tilde{\mathbf{z}}_n - \mathbf{Q}\mathbf{z}_n|| + \gamma_n ||\mathbf{c}_n|| \\ &\leq (1 + \gamma_n \beta)||\tilde{\mathbf{z}}_n - \mathbf{z}_n|| + \gamma_n ||\mathbf{c}_n|| \\ &\leq 2(||\mathbf{a}_n||/\beta + ||\mathbf{b}_n||) + ||\mathbf{c}_n||/\beta. \end{aligned} \tag{28}$$

Combining Eqs. 18, 26 and 28 yields $||\mathbf{e}_n|| \leq ||\tilde{\mathbf{y}}_n - \mathbf{y}_n|| + ||\tilde{\mathbf{q}}_n - \mathbf{q}_n|| \leq 3||\mathbf{a}_n||/\beta + 2||\mathbf{b}_n|| + ||\mathbf{c}_n||/\beta$ and, in view of Eq. 25 and Eq. 20, it follows that

$$\sum_{n \in \mathbb{N}} |||\mathbf{e}_n||| \leq \left( \max_{i=1,\ldots,m} \frac{1}{\sqrt{p_i}} \right) \sum_{n \in \mathbb{N}} ||\mathbf{e}_n|| < +\infty \quad \text{P-a.s.} \tag{29}$$

Let us set

$$\mathbf{u}_n = \gamma_n^{-1}(\mathbf{x}_n - \tilde{\mathbf{z}}_n) + \mathbf{Q}\tilde{\mathbf{z}}_n - \mathbf{Q}\mathbf{x}_n. \tag{30}$$

Note that Eq. 18 and [3, Proposition 23.2(ii)] yield

$$\tilde{\mathbf{y}}_n - \tilde{\mathbf{z}}_n \in \gamma_n \mathbf{M}\tilde{\mathbf{z}}_n \quad \text{and} \quad \mathbf{u}_n = \gamma_n^{-1}(\tilde{\mathbf{y}}_n - \tilde{\mathbf{z}}_n) + \mathbf{Q}\tilde{\mathbf{z}}_n \in \mathbf{M}\tilde{\mathbf{z}}_n + \mathbf{Q}\tilde{\mathbf{z}}_n. \tag{31}$$

Now let $\mathbf{z} \in \operatorname{zer}(\mathbf{M} + \mathbf{Q})$ and let $n \in \mathbb{N}$. We first note that

$$(\mathbf{z}, -\gamma_n \mathbf{Q}\mathbf{z}) \in \operatorname{gra} \gamma_n \mathbf{M}. \tag{32}$$

On the other hand, by Eq. 31 we have $(\tilde{\mathbf{z}}_n, \tilde{\mathbf{y}}_n - \tilde{\mathbf{z}}_n) \in \operatorname{gra} \gamma_n \mathbf{M}$. Hence, by Eq. 32 and monotonicity of $\gamma_n \mathbf{M}$, $\langle \tilde{\mathbf{z}}_n - \mathbf{z}, \tilde{\mathbf{z}}_n - \tilde{\mathbf{y}}_n - \gamma_n \mathbf{Q}\mathbf{z} \rangle \leq 0$. However, by monotonicity of $\mathbf{Q}$, $\langle \tilde{\mathbf{z}}_n - \mathbf{z}, \gamma_n \mathbf{Q}\mathbf{z} - \gamma_n \mathbf{Q}\tilde{\mathbf{z}}_n \rangle \leq 0$. Upon adding this two inequalities, we obtain

$$\langle \tilde{\mathbf{z}}_n - \mathbf{z}, \tilde{\mathbf{z}}_n - \tilde{\mathbf{y}}_n - \gamma_n \mathbf{Q}\tilde{\mathbf{z}}_n \rangle \leq 0. \tag{33}$$

In turn, we derive from Eq. 18 that

$$\begin{aligned} 2\gamma_n \langle \tilde{\mathbf{z}}_n - \mathbf{z}, \mathbf{Q}\mathbf{x}_n - \mathbf{Q}\tilde{\mathbf{z}}_n \rangle &= 2\langle \tilde{\mathbf{z}}_n - \mathbf{z}, \tilde{\mathbf{z}}_n - \tilde{\mathbf{y}}_n - \gamma_n \mathbf{Q}\tilde{\mathbf{z}}_n \rangle \\ &\quad + 2\langle \tilde{\mathbf{z}}_n - \mathbf{z}, \gamma_n \mathbf{Q}\mathbf{x}_n + \tilde{\mathbf{y}}_n - \tilde{\mathbf{z}}_n \rangle \\ &\leq 2\langle \tilde{\mathbf{z}}_n - \mathbf{z}, \gamma_n \mathbf{Q}\mathbf{x}_n + \tilde{\mathbf{y}}_n - \tilde{\mathbf{z}}_n \rangle \\ &= 2\langle \tilde{\mathbf{z}}_n - \mathbf{z}, \mathbf{x}_n - \tilde{\mathbf{z}}_n \rangle \\ &= ||\mathbf{x}_n - \mathbf{z}||^2 - ||\tilde{\mathbf{z}}_n - \mathbf{z}||^2 - ||\mathbf{x}_n - \tilde{\mathbf{z}}_n||^2. \end{aligned} \tag{34}$$

Furthermore, for $\boldsymbol{\eta} = (\eta_1, ..., \eta_m) \in \mathbf{D}$ and for all $i \in \{1, ..., m\}$ we derive from Eq. 18

$$
\begin{aligned}
&||x_{i,n} + \eta_i(q_{i,n} - y_{i,n}) - z_i||^2 \\
&\leq 1_{\{\eta_i=1\}} \big(||x_{i,n} + \tilde{q}_{i,n} - \tilde{y}_{i,n} - z_i|| + ||e_{i,n}||\big)^2 + 1_{\{\eta_i=0\}} ||x_{i,n} - z_i||^2 \\
&= 1_{\{\eta_i=1\}} \big(||(\tilde{z}_{i,n} - z_i) + \gamma_n(Q_i\mathbf{x}_n - Q_i\tilde{\mathbf{z}}_n)||^2 \\
&\quad + 2||x_{i,n} + \tilde{q}_{i,n} - \tilde{y}_{i,n} - z_i||||e_{i,n}|| + ||e_{i,n}||^2\big) + 1_{\{\eta_i=0\}} ||x_{i,n} - z_i||^2,
\end{aligned}
\tag{35}
$$

where $1_{\{\eta_i=1\}} = \left\{ \begin{array}{c} 1 \text{ if } \eta_i = 1 \\ 0 \text{ otherwise} \end{array} \right\}$ denotes the indicator function, and analogously,

$$
\begin{aligned}
&||x_{i,n} + \eta_i(\tilde{q}_{i,n} - \tilde{y}_{i,n}) - z_i||^2 = 1_{\{\eta_i=1\}} ||x_{i,n} + \tilde{q}_{i,n} - \tilde{y}_{i,n} - z_i||^2 + 1_{\{\eta_i=0\}} ||x_{i,n} - z_i||^2 \\
&\quad = 1_{\{\eta_i=1\}} ||(\tilde{z}_{i,n} - z_i) + \gamma_n(Q_i\mathbf{x}_n - Q_i\tilde{\mathbf{z}}_n)||^2 + 1_{\{\eta_i=0\}} ||x_{i,n} - z_i||^2.
\end{aligned}
\tag{36}
$$

(The Eqs. 30, 31, 33, 34, 35 and 36 hold P-surely, i.e. for all $\omega \in \Omega$.) Thus Eqs. 17, 20, 21, 22, 36, (iii), 34, the $\beta$-Lipschitz continuity of $\mathbf{Q}$ and the fact, that $0 < 1 - (1-\rho)^2 < 1$ yield

$$
\begin{aligned}
&\mathrm{E}(|||\mathbf{x}_n + \mathbf{T}_{\boldsymbol{\varepsilon}_n}(\tilde{\mathbf{q}}_n - \tilde{\mathbf{y}}_n) - \mathbf{z}|||^2 \,\big|\, \mathfrak{X}_n) = \sum_{i=0}^{m} \frac{1}{p_i} \mathrm{E}(||x_{i,n} + \varepsilon_{i,n}(\tilde{q}_{i,n} - \tilde{y}_{i,n}) - z_i||^2 \,\big|\, \mathfrak{X}_n) \\
&\quad = \sum_{i=0}^{m} \frac{1}{p_i} \sum_{\boldsymbol{\eta} \in \mathbf{D}} \mathrm{P}[\boldsymbol{\varepsilon}_n = \boldsymbol{\eta}] ||x_{i,n} + \eta_i(\tilde{q}_{i,n} - \tilde{y}_{i,n}) - z_i||^2 \\
&\quad = \sum_{i=0}^{m} \frac{1}{p_i} \Big( \sum_{\boldsymbol{\eta} \in \mathbf{D}, \eta_i = 1} \mathrm{P}[\boldsymbol{\varepsilon}_n = \boldsymbol{\eta}] |||(\tilde{z}_{i,n} - z_i) + \gamma_n(Q_i\mathbf{x}_n - Q_i\tilde{\mathbf{z}}_n)||^2 \\
&\qquad + \sum_{\boldsymbol{\eta} \in \mathbf{D}, \eta_i = 0} \mathrm{P}[\boldsymbol{\varepsilon}_n = \boldsymbol{\eta}] ||x_{i,n} - z_i||^2 \Big) \\
&\quad = ||(\tilde{\mathbf{z}}_n - \mathbf{z}) + \gamma_n(\mathbf{Q}\mathbf{x}_n - \mathbf{Q}\tilde{\mathbf{z}}_n)||^2 + \sum_{i=0}^{m} \frac{1 - p_i}{p_i} ||x_{i,n} - z_i||^2 \\
&\quad = ||\tilde{\mathbf{z}}_n - \mathbf{z}||^2 + 2\gamma_n \langle \tilde{\mathbf{z}}_n - \mathbf{z}, \mathbf{Q}\mathbf{x}_n - \mathbf{Q}\tilde{\mathbf{z}}_n \rangle + \gamma_n^2 ||\mathbf{Q}\mathbf{x}_n - \mathbf{Q}\tilde{\mathbf{z}}_n||^2 \\
&\qquad - ||\mathbf{x}_n - \mathbf{z}||^2 + |||\mathbf{x}_n - \mathbf{z}|||^2 \\
&\quad \leq ||\tilde{\mathbf{z}}_n - \mathbf{z}||^2 + ||\mathbf{x}_n - \mathbf{z}||^2 - ||\tilde{\mathbf{z}}_n - \mathbf{z}||^2 - ||\mathbf{x}_n - \tilde{\mathbf{z}}_n||^2 \\
&\qquad + \gamma_n^2 ||\mathbf{Q}\mathbf{x}_n - \mathbf{Q}\tilde{\mathbf{z}}_n||^2 - ||\mathbf{x}_n - \mathbf{z}||^2 + |||\mathbf{x}_n - \mathbf{z}|||^2 \\
&\quad \leq -\big(1 - (\gamma_n\beta)^2\big) ||\mathbf{x}_n - \tilde{\mathbf{z}}_n||^2 + |||\mathbf{x}_n - \mathbf{z}|||^2 \\
&\quad \leq -\big(1 - (1-\rho)^2\big) ||\mathbf{x}_n - \tilde{\mathbf{z}}_n||^2 + |||\mathbf{x}_n - \mathbf{z}|||^2 \leq |||\mathbf{x}_n - \mathbf{z}|||^2 \qquad \text{P-a. s.}
\end{aligned}
\tag{37}
$$

Thus it follows from the Eqs. 18, 19, the Jensen's inequality, Eq. 37 and the fact, that, since $\mathbf{e}_n = \gamma_n(\mathbf{a}_n - \mathbf{c}_n)$ and $\mathbf{a}_n$ and $\mathbf{c}_n$ are $\mathfrak{X}_n$-measurable, $|||\mathbf{e}_n|||$ is a $\mathfrak{X}_n$-measurable random variable

$$
\begin{aligned}
\mathrm{E}(|||\mathbf{x}_{n+1} - \mathbf{z}||| \,\big|\, \mathfrak{X}_n) &\leq \mathrm{E}(|||\mathbf{x}_n + \mathbf{T}_{\boldsymbol{\varepsilon}_n}(\tilde{\mathbf{q}}_n - \tilde{\mathbf{y}}_n) - \mathbf{z}||| \,\big|\, \mathfrak{X}_n) + \mathrm{E}(|||\tilde{\mathbf{e}}_n||| \,\big|\, \mathfrak{X}_n) \\
&\leq \sqrt{\mathrm{E}(|||\mathbf{x}_n + \mathbf{T}_{\boldsymbol{\varepsilon}_n}(\tilde{\mathbf{q}}_n - \tilde{\mathbf{y}}_n) - \mathbf{z}|||^2 \,\big|\, \mathfrak{X}_n)} + \mathrm{E}(|||\mathbf{e}_n||| \,\big|\, \mathfrak{X}_n) \\
&\leq |||\mathbf{x}_n - \mathbf{z}||| + |||\mathbf{e}_n||| \qquad \text{P-a. s.}
\end{aligned}
\tag{38}
$$

Altogether, inequality 14 of Theorem 1 is satisfied with $\vartheta_n(\mathbf{z}) = \mathbf{0}$, $\chi_n(\mathbf{z}) = \mathbf{0}$, $\eta_n(\mathbf{z}) = |||\mathbf{e}_n|||$, where $(|||\mathbf{e}_n|||)_{n\in\mathbb{N}} \in \ell_+^1(\mathscr{X})$ due to Eq. 29, $\phi(x) = x$ and $F = \mathrm{zer}(\mathbf{M} + \mathbf{Q})$,

which is closed by [3, Proposition 20.33 (iii)] since $\mathbf{M}+\mathbf{Q}$ is maximally monotone (Eq. 24) we have $w \in \operatorname{zer}(\mathbf{M}+\mathbf{Q}) \iff (w,0) \in \operatorname{gra}(\mathbf{M}+\mathbf{Q})$. We therefore derive from Theorem 1 (ii), that $(\mathbf{x}_n)_{n\in\mathbb{N}}$ is bounded P-a. s. with respect to $|||\cdot|||$, and we deduce from Eq. 18 that, since the norms $|||\cdot|||$ and $||\cdot||$ are equivalent and since the operators $\mathbf{Q}$ and $\mathbf{J}_{\gamma_n\mathbf{M}}$ are Lipschitzian, $(\tilde{\mathbf{y}}_n)_{n\in\mathbb{N}}$, $(\tilde{\mathbf{z}}_n)_{n\in\mathbb{N}}$ and $(\tilde{\mathbf{q}}_n)_{n\in\mathbb{N}}$ are bounded P-a. s. with respect to $||\cdot||$, hence for all $\mathbf{z} \in \operatorname{zer}(\mathbf{M}+\mathbf{Q})$ exists a $\mu_{\mathbf{z}} \in \mathbb{R}$ such that for all $i \in \{1,...,m\}$ and all $n \in \mathbb{N}$

$$||x_{i,n} + \tilde{q}_{i,n} - \tilde{y}_{i,n} - z_i|| \leq \mu_{\mathbf{z}} \quad \text{P-a. s.} \tag{39}$$

Now we proceed similarly as we did in Eq. 37, thus Eqs. 20, 21, 23, 35, (iii), 39, 34, the $\beta$-Lipschitz continuity of $\mathbf{Q}$ and the fact, that $0 < 1-(1-\rho)^2 < 1$ yield

$$\begin{aligned}
\mathrm{E}(|||\mathbf{x}_{n+1}-\mathbf{z}|||^2 \,|\, \mathfrak{X}_n) &= \sum_{i=0}^{m} \frac{1}{p_i} \mathrm{E}(||x_{i,n+1}-z_i||^2 \,|\, \mathfrak{X}_n) \\
&= \sum_{i=0}^{m} \frac{1}{p_i} \sum_{\boldsymbol{\eta}\in\mathbf{D}} \mathrm{P}[\boldsymbol{\varepsilon}_n = \boldsymbol{\eta}] ||x_{i,n} + \eta_i(q_{i,n}-y_{i,n}) - z_i||^2 \\
&\leq \sum_{i=0}^{m} \frac{1}{p_i} \Big( \sum_{\boldsymbol{\eta}\in\mathbf{D},\eta_i=1} \mathrm{P}[\boldsymbol{\varepsilon}_n = \boldsymbol{\eta}] \big( ||(\tilde{z}_{i,n}-z_i) + \gamma_n(Q_i\mathbf{x}_n - Q_i\tilde{\mathbf{z}}_n)||^2 + \\
&\quad + 2||x_{i,n}+\tilde{q}_{i,n}-\tilde{y}_{i,n}-z_i||\ ||e_{i,n}|| + ||e_{i,n}||^2 \big) \\
&\quad + \sum_{\boldsymbol{\eta}\in\mathbf{D},\eta_i=0} \mathrm{P}[\boldsymbol{\varepsilon}_n = \boldsymbol{\eta}] ||x_{i,n}-z_i||^2 \Big) \\
&= ||(\tilde{\mathbf{z}}_n - \mathbf{z}) + \gamma_n(\mathbf{Q}\mathbf{x}_n - \mathbf{Q}\tilde{\mathbf{z}}_n)||^2 + ||\mathbf{e}_n||^2 \\
&\quad + 2\sum_{i=0}^{m} ||x_{i,n}+\tilde{q}_{i,n}-\tilde{y}_{i,n}-z_i||\ ||e_{i,n}|| + \sum_{i=0}^{m} \frac{1-p_i}{p_i} ||x_{i,n}-z_i||^2 \\
&\leq ||\tilde{\mathbf{z}}_n - \mathbf{z}||^2 + 2\gamma_n \langle \tilde{\mathbf{z}}_n - \mathbf{z}, \mathbf{Q}\mathbf{x}_n - \mathbf{Q}\tilde{\mathbf{z}}_n \rangle + \gamma_n^2 ||\mathbf{Q}\mathbf{x}_n - \mathbf{Q}\tilde{\mathbf{z}}_n||^2 \\
&\quad + 2\mu_{\mathbf{z}}||\mathbf{e}_n|| + ||\mathbf{e}_n||^2 - ||\mathbf{x}_n - \mathbf{z}||^2 + |||\mathbf{x}_n - \mathbf{z}|||^2 \\
&\leq ||\tilde{\mathbf{z}}_n - \mathbf{z}||^2 + ||\mathbf{x}_n - \mathbf{z}||^2 - ||\tilde{\mathbf{z}}_n - \mathbf{z}||^2 - ||\mathbf{x}_n - \tilde{\mathbf{z}}_n||^2 + \gamma_n^2 ||\mathbf{Q}\mathbf{x}_n - \mathbf{Q}\tilde{\mathbf{z}}_n||^2 \\
&\quad + 2\mu_{\mathbf{z}}||\mathbf{e}_n|| + ||\mathbf{e}_n||^2 - ||\mathbf{x}_n - \mathbf{z}||^2 + |||\mathbf{x}_n - \mathbf{z}|||^2 \\
&\leq -\big(1-(1-\rho)^2\big) ||\mathbf{x}_n - \tilde{\mathbf{z}}_n||^2 + |||\mathbf{x}_n - \mathbf{z}|||^2 + 2\mu_{\mathbf{z}}||\mathbf{e}_n|| + ||\mathbf{e}_n||^2 \quad \text{P-a. s.}
\end{aligned} \tag{40}$$

Altogether, inequality 14 of Theorem 1 is satisfied with $\vartheta_n(\mathbf{z}) = \big(1-(1-\rho)^2\big)||\mathbf{x}_n - \tilde{\mathbf{z}}_n||^2$, where $\big(\big(1-(1-\rho)^2\big)||\mathbf{x}_n - \tilde{\mathbf{z}}_n||^2\big)_{n\in\mathbb{N}} \in \ell_+(\mathscr{X})$, $\chi_n(\mathbf{z}) = \mathbf{0}$, $\eta_n(\mathbf{z}) = 2\mu_{\mathbf{z}}||\mathbf{e}_n|| + ||\mathbf{e}_n||^2$, where $(2\mu_{\mathbf{z}}||\mathbf{e}_n|| + ||\mathbf{e}_n||^2)_{n\in\mathbb{N}} \in \ell_+^1(\mathscr{H})$ due to Eq. 29, $\phi(x) = x^2$ and $F = \operatorname{zer}(\mathbf{M}+\mathbf{Q})$. We therefore derive from Theorem 1 (i) that $\forall \mathbf{z} \in \operatorname{zer}(\mathbf{M}+\mathbf{Q})\ \ \sum_{n\in\mathbb{N}} \vartheta_n(\mathbf{z}) < +\infty$ P-a. s., this yields

$$\sum_{n\in\mathbb{N}} ||\mathbf{x}_n - \tilde{\mathbf{z}}_n||^2 < +\infty \quad \text{P-a. s.} \tag{41}$$

Hence, since (i) and Eq. 27 imply $\sum_{n\in\mathbb{N}} ||\tilde{\mathbf{z}}_n - \mathbf{z}_n|| < +\infty$ P-a. s., we have $\sum_{n\in\mathbb{N}} ||\tilde{\mathbf{z}}_n - \mathbf{z}_n||^2 < +\infty$ P-a. s. We therefore infer that $\sum_{n\in\mathbb{N}} ||\mathbf{x}_n - \mathbf{z}_n||^2 < +\infty$ P-a. s., which proves (a).

(b): It follows from Eq. 41, the Lipschitz continuity of $\mathbf{Q}$ and Eq. 30 that

$$\mathbf{Q}\mathbf{x}_n - \mathbf{Q}\tilde{\mathbf{z}}_n \to \mathbf{0} \quad \text{P-a. s.} \quad \text{and} \quad \mathbf{u}_n \to \mathbf{0} \quad \text{P-a. s.} \tag{42}$$

Now we show $\mathcal{W}(\mathbf{x}_n)_{n\in\mathbb{N}} \subset \operatorname{zer}(\mathbf{M}+\mathbf{Q})$ P-a. s. Let $\mathbf{x}$ be a weak sequential cluster point of $(\mathbf{x}_n)_{n\in\mathbb{N}}$, say $\mathbf{x}_{k_n} \rightharpoonup \mathbf{x}$. It remains to show that $(\mathbf{x},\mathbf{0}) \in \operatorname{gra}(\mathbf{M}+\mathbf{Q})$ P-a. s. It follows from Eq. 41 that $\tilde{\mathbf{z}}_{k_n} \rightharpoonup \mathbf{x}$ P-a. s. and from Eq. 42 that $\mathbf{u}_{k_n} \to \mathbf{0}$ P-a. s. Altogether, there exists a $\tilde{\Omega} \subset \Omega$ with $P(\tilde{\Omega}) = 1$ such that for all $\omega \in \tilde{\Omega}$ we have $(\tilde{\mathbf{z}}_{k_n}(\omega), \mathbf{u}_{k_n}(\omega)) \in \operatorname{gra}(\mathbf{M}+\mathbf{Q})$ by Eq. 31, and it satisfies

$$\tilde{\mathbf{z}}_{k_n}(\omega) \rightharpoonup \mathbf{x}(\omega) \qquad \text{and} \qquad \mathbf{u}_{k_n}(\omega) \to \mathbf{0}. \tag{43}$$

Since $\mathbf{M}+\mathbf{Q}$ is maximally monotone (Eq. 24), it follows from [3, Proposition 20.33(ii)], that $(\mathbf{x}(\omega),\mathbf{0}) \in \operatorname{gra}(\mathbf{M}+\mathbf{Q})$ for all $\omega \in \tilde{\Omega}$. We showed $\mathcal{W}(\mathbf{x}_n)_{n\in\mathbb{N}} \subset \operatorname{zer}(\mathbf{M}+\mathbf{Q})$ P-a. s., this together with the passage under Eq. 40 allows us to use Theorem 1 (iv), which implies $(\mathbf{x}_n)_{n\in\mathbb{N}}$ converge weakly P-a. s. to a $\operatorname{zer}(\mathbf{M}+\mathbf{Q})$-valued random variable $\overline{\mathbf{x}}$. This together with (a) implies $(\mathbf{z}_n)_{n\in\mathbb{N}}$ converge weakly P-a. s. to $\overline{\mathbf{x}}$. This proves (b).

(c): The assumptions in (c) imply that $\mathbf{M}+\mathbf{Q}$ is uniformly monotone at $\overline{\mathbf{x}}$. Hence, since $\overline{\mathbf{x}} \in \operatorname{zer}(\mathbf{M}+\mathbf{Q}) \iff (\overline{\mathbf{x}},\mathbf{0}) \in \operatorname{gra}(\mathbf{M}+\mathbf{Q})$, it follows from Eq. 31 und 7 that there exists an increasing function $\phi : [0,+\infty[ \to [0,+\infty]$ that vanishes only at 0 such that

$$(\forall n \in \mathbb{N}) \qquad \langle \tilde{\mathbf{z}}_n - \mathbf{z}_n, \mathbf{u}_n \rangle + \langle \mathbf{z}_n - \overline{\mathbf{x}}, \mathbf{u}_n \rangle = \langle \tilde{\mathbf{z}}_n - \overline{\mathbf{x}}, \mathbf{u}_n \rangle \geq \phi(||\tilde{\mathbf{z}}_n - \overline{\mathbf{x}}||). \tag{44}$$

We therefore deduce from Eq. 42, 27, (i), (b) and [3, Lemma 2.41(iii)] that $\phi(||\tilde{\mathbf{z}}_n - \overline{\mathbf{x}}||) \to 0$ P-a. s., which implies $\tilde{\mathbf{z}}_n \to \overline{\mathbf{x}}$ P-a. s., due to Eq. 27 and (i) we derive $\mathbf{z}_n \to \overline{\mathbf{x}}$ P-a. s. In turn, (a) yields $\mathbf{x}_n \to \overline{\mathbf{x}}$ P-a. s.

□

Claim (c) of Theorem 3 is not used anymore in this thesis, however, we proved it for the sake of completeness. Some remarks about the structure of Algorithm 2 and the rule of the arising stochastic variables $\varepsilon_i$ are given afterwards, first we do the proof of the main result.

*Proof to Theorem 2.* (The paradigm to this proof is the proof of [2, Theorem 3.1].) We are going to transform Algorihm 1 equivalently into Algorithm 2 of Theorem 3, then we verify its conditions, this allows us to apply it and therefore proves the essential part of Theorem 2. Let us first rewrite Algorihm 1 in three steps. In the first step we dissolve the two inner **if**-loops in line 6 to 12 and in line 21 to 25, respectively, of Algorithm 1. Moreover, we write the two **for**-loops in line 15 and 20 of Algorithm 1 as one common **for**-loop. The result looks as follows.

**Algorithm 3:**

**for** $n = 0, 1, \ldots$ **do**
  $y_{1,n} = x_n - \gamma_n(Cx_n + \sum_{i=1}^m L_i^* v_{i,n} + a_{1,n})$
  $p_{1,n} = J_{\gamma_n A}(y_{1,n}) + b_{1,n}$
  **if** $\varepsilon_{1,n} = 0$ **then**
    **for** $i = 1, \ldots, m$ **do**
      $y_{2,i,n} = v_{i,n} + \gamma_n(L_i x_n - D_i^{-1} v_{i,n} - a_{2,i,n})$
      $p_{2,i,n} = J_{\gamma_n B_i^{-1}}(y_{2,i,n}) + b_{2,i,n}$
      $q_{2,i,n} = p_{2,i,n} + \gamma_n(L_i p_{1,n} - D_i^{-1} p_{2,i,n} - c_{2,i,n})$
      $v_{i,n+1} = v_{i,n} - \varepsilon_{2,i,n}(y_{2,i,n} + q_{2,i,n})$
    $x_{n+1} = x_n$
  **else** $[\varepsilon_{1,n} = 1]$
    **for** $i = 1, \ldots, m$ **do**
      $y_{2,i,n} = v_{i,n} + \gamma_n(L_i x_n - D_i^{-1} v_{i,n} - a_{2,i,n})$
      $p_{2,i,n} = J_{\gamma_n B_i^{-1}}(y_{2,i,n}) + b_{2,i,n}$
      $q_{2,i,n} = p_{2,i,n} + \gamma_n(L_i p_{1,n} - D_i^{-1} p_{2,i,n} - c_{2,i,n})$
      $v_{i,n+1} = v_{i,n} - \varepsilon_{2,i,n}(y_{2,i,n} + q_{2,i,n})$
    $q_{1,n} = p_{1,n} - \gamma_n(Cp_{1,n} + \sum_{i=1}^m L_i^* p_{2,i,n} + c_{1,n})$
    $x_{n+1} = x_n - y_{1,n} + q_{1,n}$

In the second step we dissolve the remaining **if**-loop and unite the two **for**-loops of Algorithm 3, the resulting Algorithm reads as follows.

**Algorithm 4:**

**for** $n = 0, 1, \ldots$ **do**
  $y_{1,n} = x_n - \gamma_n(Cx_n + \sum_{i=1}^m L_i^* v_{i,n} + a_{1,n})$
  $p_{1,n} = J_{\gamma_n A}(y_{1,n}) + b_{1,n}$
  **for** $i = 1, \ldots, m$ **do**
    $y_{2,i,n} = v_{i,n} + \gamma_n(L_i x_n - D_i^{-1} v_{i,n} - a_{2,i,n})$
    $p_{2,i,n} = J_{\gamma_n B_i^{-1}}(y_{2,i,n}) + b_{2,i,n}$
    $q_{2,i,n} = p_{2,i,n} + \gamma_n(L_i p_{1,n} - D_i^{-1} p_{2,i,n} - c_{2,i,n})$
    $v_{i,n+1} = v_{i,n} - \varepsilon_{2,i,n}(y_{2,i,n} + q_{2,i,n})$
  $q_{1,n} = p_{1,n} - \gamma_n(Cp_{1,n} + \sum_{i=1}^m L_i^* p_{2,i,n} + c_{1,n})$
  $x_{n+1} = x_n - \varepsilon_{1,n}(y_{1,n} + q_{1,n})$

In the last step we just change the structure by putting the equally denoted variables together, the result looks as follows.

**Algorithm 5:**

**for** $n = 0, 1, \ldots$ **do**
  $y_{1,n} = x_n - \gamma_n(Cx_n + \sum_{i=1}^m L_i^* v_{i,n} + a_{1,n})$
  **for** $i = 1, \ldots, m$ **do**
    $y_{2,i,n} = v_{i,n} + \gamma_n(L_i x_n - D_i^{-1} v_{i,n} - a_{2,i,n})$
  $p_{1,n} = J_{\gamma_n A}(y_{1,n}) + b_{1,n}$
  **for** $i = 1, \ldots, m$ **do**
    $p_{2,i,n} = J_{\gamma_n B_i^{-1}}(y_{2,i,n}) + b_{2,i,n}$
  $q_{1,n} = p_{1,n} - \gamma_n(Cp_{1,n} + \sum_{i=1}^m L_i^* p_{2,i,n} + c_{1,n})$
  **for** $i = 1, \ldots, m$ **do**
    $q_{2,i,n} = p_{2,i,n} + \gamma_n(L_i p_{1,n} - D^{-1} p_{2,i,n} - c_{2,i,n})$
  $x_{n+1} = x_n - \varepsilon_{1,n}(y_{1,n} + q_{1,n})$
  **for** $i = 1, \ldots, m$ **do**
    $v_{i,n+1} = v_{i,n} - \varepsilon_{2,i,n}(y_{2,i,n} + q_{2,i,n})$

The Algorithms 3, 4 and 5 are all equivalent to Algorithm 1, but the computational effort is different (see Remark 4(i)).

Next, we introduce the Hilbert space

$$\boldsymbol{\mathcal{K}} = \mathcal{H} \oplus \mathcal{G}_1 \oplus \cdots \oplus \mathcal{G}_m, \tag{45}$$

and the operators

$$\begin{aligned} \mathbf{M} : \boldsymbol{\mathcal{K}} &\to 2^{\boldsymbol{\mathcal{K}}} \\ (x, v_1, \ldots, v_m) &\mapsto Ax \times B_1^{-1} v_1 \times \cdots \times B_m^{-1} v_m \end{aligned} \tag{46}$$

and

$$\begin{aligned} \mathbf{Q} : \boldsymbol{\mathcal{K}} &\to \boldsymbol{\mathcal{K}} \\ (x, v_1, \ldots, v_m) &\mapsto (Cx + L_1^* v_1 + \cdots + L_m^* v_m, -L_1 x + D_1^{-1} v_1, \ldots, -L_m x + D_m^{-1} v_m). \end{aligned} \tag{47}$$

Since the operators $A$ and $(B_i)_{1 \leq i \leq m}$ are maximally monotone, so is $\mathbf{M}$ maximally monotone by [3, Propositions 20.22 and 20.23]. Additionally, we derive from [3, Propositions 23.16]

$$\begin{aligned} &(\forall \gamma \in ]0, +\infty[\,)(\forall (x, v_1, \ldots, v_m) \in \boldsymbol{\mathcal{K}}) \\ &\mathbf{J}_{\gamma \mathbf{M}}(x, v_1, \ldots, v_m) = \big(J_{\gamma A}(x), J_{\gamma B_1^{-1}}(v_1), \ldots, J_{\gamma B_m^{-1}}(v_m)\big). \end{aligned} \tag{48}$$

Now we examine the properties of $\mathbf{Q}$. Therefore, let $(x, v_1, \ldots, v_m)$ and $(y, w_1, \ldots, w_m)$ be two points in $\boldsymbol{\mathcal{K}}$. The monotonicity of the operators $C$ and $(D_i^{-1})_{1 \leq i \leq m}$ and Eq. 47

yield

$$
\begin{aligned}
&\langle (x,v_1,...,v_m)-(y,w_1,...,w_m), \mathbf{Q}(x,v_1,...,v_m)-\mathbf{Q}(y,w_1,...,w_m)\rangle \\
&\quad = \big\langle (x-y, v_1-w_1,...,v_m-w_m), \big(Cx-Cy+L_1^*(v_1-w_1) \\
&\qquad +\cdots+L_m^*(v_m-w_m), -L_1(x-y)+D_1^{-1}v_1-D_1^{-1}w_1,..., \\
&\qquad -L_m(x-y)+D_m^{-1}v_m-D_m^{-1}w_m\big)\big\rangle \\
&\quad = \langle x-y, Cx-Cy\rangle + \sum_{i=1}^m \langle v_i-w_i, D_i^{-1}v_i-D_i^{-1}w_i\rangle \\
&\qquad + \sum_{i=1}^m \big(\big\langle x-y, L_i^*(v_i-w_i)\big\rangle - \langle v_i-w_i, L_i(x-y)\rangle\big) \\
&\quad = \langle x-y, Cx-Cy\rangle + \sum_{i=1}^m \langle v_i-w_i, D_i^{-1}v_i-D_i^{-1}w_i\rangle \\
&\quad \geq 0.
\end{aligned} \tag{49}
$$

Hence, $\mathbf{Q}$ is monotone. The triangle inequality, the Lipschitzianity assumptions, the Cauchy-Schwarz inequality, and Eq. 16 yield

$$
\begin{aligned}
&||\mathbf{Q}(x,v_1,...,v_m)-\mathbf{Q}(y,w_1,...,w_m)|| \\
&\quad = \Big|\Big|\Big(Cx-Cy, D_1^{-1}v_1-D_1^{-1}w_1,..., D_m^{-1}v_m-D_m^{-1}w_m\Big) \\
&\qquad + \Big(\sum_{i=1}^m L_i^*(v_i-w_i), -L_1(x-y),..., -L_m(x-y)\Big)\Big|\Big| \\
&\quad \leq \Big|\Big|\Big(Cx-Cy, D_1^{-1}v_1-D_1^{-1}w_1,..., D_m^{-1}v_m-D_m^{-1}w_m\Big)\Big|\Big| \\
&\qquad + \Big|\Big|\Big(\sum_{i=1}^m L_i^*(v_i-w_i), -L_1(x-y),..., -L_m(x-y)\Big)\Big|\Big| \\
&\quad = \sqrt{||Cx-Cy||^2+\sum_{i=1}^m \big|\big|D_i^{-1}v_i-D_i^{-1}w_i\big|\big|^2} \\
&\qquad + \sqrt{\Big|\Big|\sum_{i=1}^m L_i^*(v_i-w_i)\Big|\Big|^2+\sum_{i=1}^m ||L_i(x-y)||^2} \\
&\quad \leq \sqrt{\mu^2||x-y||^2+\sum_{i=1}^m \nu_i^2||v_i-w_i||^2} \\
&\qquad + \sqrt{\Big(\sum_{i=1}^m ||L_i||||v_i-w_i||\Big)^2+\sum_{i=1}^m ||L_i||^2||x-y||^2} \\
&\quad \leq \max\{\mu,\nu_1,...,\nu_m\}\sqrt{||x-y||^2+\sum_{i=1}^m ||v_i-w_i||^2} \\
&\qquad + \sqrt{\Big(\sum_{i=1}^m ||L_i||^2\Big)\Big(\sum_{i=1}^m ||v_i-w_i||^2\Big)+\Big(\sum_{i=1}^m ||L_i||^2\Big)||x-y||^2} \\
&\quad = \beta||(x,v_1,...,v_m)-(y,w_1,...,w_m)||.
\end{aligned} \tag{50}
$$

After all, we have shown that

(51) $\mathbf{M}$ is maximally monotone and $\mathbf{Q}$ is monotone and $\beta$-Lipschitzian.

Now, we observe that

(52)

$$
\begin{aligned}
\text{Eq. 15} &\iff (\exists x \in \mathcal{H})\ \ 0 \in Ax + \sum_{i=1}^{m} L_i^* \big((B_i \ \square \ D_i)(L_i x)\big) + Cx \\
&\iff (\exists (x, v_1, ..., v_m) \in \mathfrak{K}) \begin{cases} 0 \in Ax + \sum_{i=1}^{m} L_i^* v_i + Cx \\ v_1 \in (B_1 \ \square \ D_1)(L_1 x) \\ . \\ . \\ . \\ v_m \in (B_m \ \square \ D_m)(L_m x) \end{cases} \\
&\iff (\exists (x, v_1, ..., v_m) \in \mathfrak{K}) \begin{cases} 0 \in Ax + \sum_{i=1}^{m} L_i^* v_i + Cx \\ 0 \in B_1^{-1} v_1 + D_1^{-1} v_1 - L_1 x \\ . \\ . \\ . \\ 0 \in B_m^{-1} v_m + D_m^{-1} v_m - L_m x \end{cases} \\
&\iff (\exists (x, v_1, ..., v_m) \in \mathfrak{K})\ (0, ..., 0) \in Ax \times B_1^{-1} v_1 \times \cdots \times B_m^{-1} v_m \\
&\qquad\qquad + (L_1^* v_1 + \cdots + L_m^* v_m + Cx, \\
&\qquad\qquad\quad D_1^{-1} v_1 - L_1 x, ..., D_m^{-1} v_m - L_m x) \\
&\iff (\exists (x, v_1, ..., v_m) \in \mathfrak{K})\ (0, ..., 0) \in (\mathbf{M} + \mathbf{Q})(x, v_1, ..., v_m).
\end{aligned}
$$

Equivalently,

(53)
$$\operatorname{zer}(\mathbf{M} + \mathbf{Q}) \neq \emptyset.$$

Next, we define the random variables

(54)

$$
\forall n \in \mathbb{N} \begin{cases} \mathbf{x}_n = (x_n, v_{1,n}, ..., v_{m,n}) \\ \mathbf{y}_n = (y_{1,n}, y_{2,1,n}, ..., y_{2,m,n}) \\ \mathbf{z}_n = (p_{1,n}, p_{2,1,n}, ..., p_{2,m,n}) \\ \mathbf{q}_n = (q_{1,n}, q_{2,1,n}, ..., q_{2,m,n}) \end{cases} \text{ and } \begin{cases} \mathbf{a}_n = (a_{1,n}, a_{2,1,n}, ..., a_{2,m,n}) \\ \mathbf{b}_n = (b_{1,n}, b_{2,1,n}, ..., b_{2,m,n}) \\ \mathbf{c}_n = (c_{1,n}, c_{2,1,n}, ..., c_{2,m,n}) \\ \boldsymbol{\varepsilon}_n = (\varepsilon_{1,n}, \varepsilon_{2,1,n}, ..., \varepsilon_{2,m,n}) \end{cases}
$$

Let us define the $\sigma$-algebra $\mathfrak{X}_n = \sigma(\mathcal{X}_{1,n}, \mathcal{X}_{2,1,n}, ..., \mathcal{X}_{2,m,n})$, which is equal to $\sigma(\mathbf{x}_0, ..., \mathbf{x}_n)$. Thus and due to the assumptions in (A), we obtain that the random variables

(55)

$\mathbf{a}_n, \mathbf{b}_n$ and $\mathbf{c}_n$ are $\mathfrak{X}_n$-measurable and

$$\sum_{n \in \mathbb{N}} \mathrm{E}(||\mathbf{a}_n|| \,|\, \mathfrak{X}_n) < +\infty, \quad \sum_{n \in \mathbb{N}} \mathrm{E}(||\mathbf{b}_n|| \,|\, \mathfrak{X}_n) < +\infty \quad \text{and} \quad \sum_{n \in \mathbb{N}} \mathrm{E}(||\mathbf{c}_n|| \,|\, \mathfrak{X}_n) < +\infty.$$

Furthermore, it follows from the Eqs. 47, 48 and 54, that Algorithm 5 assumes in $\mathfrak{K}$ the form of Algorithm 2 in Theorem 3, the $m$ in Algorithm 2 corresponds to $m+1$ here.

**for** $n = 0, 1, \ldots$ **do**
  $\mathbf{y}_n = \mathbf{x}_n - \gamma_n(\mathbf{Q}\mathbf{x}_n + \mathbf{a}_n)$
  $\mathbf{z}_n = \mathbf{J}_{\gamma_n \mathbf{M}}(\mathbf{y}_n) + \mathbf{b}_n$
  $\mathbf{q}_n = \mathbf{z}_n - \gamma_n(\mathbf{Q}\mathbf{z}_n + \mathbf{c}_n)$
  **for** $i = 1, \ldots, (m+1)$ **do**
    $(\mathbf{x}_{n+1})_i = (\mathbf{x}_n)_i + (\boldsymbol{\varepsilon}_n)_i \big[(\mathbf{q}_n)_i - (\mathbf{y}_n)_i\big]$
  $\mathbf{x}_{n+1} = \big((\mathbf{x}_{n+1})_1, \ldots, (\mathbf{x}_{n+1})_{m+1}\big)$.

Now we apply Theorem 3, the Eqs. 51, 53, 55, (B), (C) and the above algorithm ensure that all of its conditions are satisfied. Moreover, its statement (a) implies that $\sum_{n\in\mathbb{N}} ||\mathbf{x}_n - \mathbf{z}_n||^2 < +\infty$ P-a. s., that proves (i).

Next we prove (ii). It follows from Theorem 3 (b) that there exists a $\overline{\mathbf{x}} \in \operatorname{zer}(\mathbf{M}+\mathbf{Q})$ such that

$$\mathbf{x}_n \rightharpoonup \overline{\mathbf{x}} \text{ P-a. s. and } \mathbf{z}_n \rightharpoonup \overline{\mathbf{x}} \text{ P-a. s.} \tag{56}$$

We set

$$\overline{\mathbf{x}} = (\overline{x}, \overline{v}_1, \ldots, \overline{v}_m). \tag{57}$$

In view of Eqs. 46 and 47,

$$\overline{\mathbf{x}} \in \operatorname{zer}(\mathbf{M}+\mathbf{Q}) \iff \begin{cases} 0 \in A\overline{x} + \sum_{i=1}^m L_i^* \overline{v}_i + C\overline{x} \\ 0 \in B_1^{-1}\overline{v}_1 + D_1^{-1}\overline{v}_1 - L_1\overline{x} \\ \vdots \\ 0 \in B_m^{-1}\overline{v}_m + D_m^{-1}\overline{v}_m - L_m\overline{x} \end{cases}$$

$$\iff \begin{cases} -\sum_{i=1}^m L_i^* \overline{v}_i \in A\overline{x} + C\overline{x} \\ L_1\overline{x} \in (B_1^{-1} + D_1^{-1})\overline{v}_1 \\ \vdots \\ L_m\overline{x} \in (B_m^{-1} + D_m^{-1})\overline{v}_m \end{cases} \tag{58}$$

$$\iff \begin{cases} -\sum_{i=1}^m L_i^* \overline{v}_i \in A\overline{x} + C\overline{x} \\ \overline{v}_1 \in (B_1 \ \square \ D_1)(L_1\overline{x}) \\ \vdots \\ \overline{v}_m \in (B_m \ \square \ D_m)(L_m\overline{x}) \end{cases} \tag{59}$$

$$\implies \begin{cases} -\sum_{i=1}^m L_i^* \overline{v}_i \in A\overline{x} + C\overline{x} \\ L_1^*\overline{v}_1 \in L_1^*(B_1 \ \square \ D_1)(L_1\overline{x}) \\ \vdots \\ L_m^*\overline{v}_m \in L_m^*(B_m \ \square \ D_m)(L_m\overline{x}) \end{cases}$$

$$\implies 0 \in A\overline{x} + \sum_{i=1}^m L_i^* \big((B_i \ \square \ D_i)(L_i\overline{x})\big) + C\overline{x}$$

$$\iff \overline{x} \text{ solves Eq. 1.} \tag{60}$$

Additionally, Eq. 59 means that

$$(\overline{v}_1, \ldots, \overline{v}_m) \text{ solves Eq. 2.} \tag{61}$$

Further, Eq. 58 implies (ii)(a).

Next we deduce from Eq. 58 that

$$\begin{aligned} &\bar{x} \in (A+C)^{-1}\Big(-\sum_{j=1}^{m} L_j^* \bar{v}_j\Big) \text{ and} \\ &(\forall i \in \{1,...,m\}) \;\; L_i \bar{x} \in (B_i^{-1} + D_i^{-1})\bar{v}_i. \end{aligned} \tag{62}$$

Hence,

$$(\forall i \in \{1,...,m\}) \quad \begin{cases} -L_i \bar{x} \in -L_i\big((A^{-1} \;\square\; C^{-1})(-\sum_{j=1}^{m} L_j^* \bar{v}_j)\big) \\ L_i \bar{x} \in (B_i^{-1} + D_i^{-1})\bar{v}_i. \end{cases} \tag{63}$$

Thus,

$$(\forall i \in \{1,...,m\}) \quad 0 \in -L_i\Big((A^{-1} \;\square\; C^{-1})\Big(-\sum_{j=1}^{m} L_j^* \bar{v}_j\Big)\Big) + B_i^{-1}\bar{v}_i + D_i^{-1}\bar{v}_i, \tag{64}$$

which proves (ii)(b). Furthermore, (ii)(c) follows from Eqs. 56, 57 and 60 and (ii)(d) follows from Eqs. 56, 57 and 61.

It remains to prove strong convergence, i.e. (ii)(e) and (ii)(f). Now we define

$$(\forall n \in \mathbb{N}) \begin{cases} \tilde{y}_{1,n} = x_n - \gamma_n(Cx_n + \sum_{j=1}^{m} L_j^* v_{j,n}) \\ \tilde{p}_{1,n} = J_{\gamma_n A}(\tilde{y}_{1,n}) \end{cases} \tag{65}$$

and

$$(\forall i \in \{1,...,m\})(\forall n \in \mathbb{N}) \begin{cases} \tilde{y}_{2,i,n} = v_{i,n} + \gamma_n(L_i x_n - D_i^{-1} v_{i,n}) \\ \tilde{p}_{2,i,n} = J_{\gamma_n B_i^{-1}}(\tilde{y}_{2,i,n}). \end{cases} \tag{66}$$

Then, due to the nonexpansiveness of the resolvents [3, Proposition 23.7] and Algorithm 5 (which is equivalent to Alg. 1, but Alg. 5 is better readable) we obtain

$$\begin{aligned} (\forall n \in \mathbb{N}) \;\; ||p_{1,n} - \tilde{p}_{1,n}|| &\leq ||J_{\gamma_n A}(y_{1,n}) + b_{1,n} - J_{\gamma_n A}(\tilde{y}_{1,n})|| \\ &\leq ||y_{1,n} - \tilde{y}_{1,n}|| + ||b_{1,n}|| \leq \gamma_n ||a_{1,n}|| + ||b_{1,n}|| \\ &\leq \beta^{-1}||a_{1,n}|| + ||b_{1,n}||. \end{aligned} \tag{67}$$

The condition (A) implies that the sequences $(a_{1,n})_{n\in\mathbb{N}}$ and $(b_{1,n})_{n\in\mathbb{N}}$ are P-a. s. absolutely summable, thus, it follows

$$y_{1,n} - \tilde{y}_{1,n} \to 0 \;\; \text{P-a. s.} \quad \text{and} \quad p_{1,n} - \tilde{p}_{1,n} \to 0 \;\; \text{P-a. s.} \tag{68}$$

Analogously, we derive from Eq. 66 and Algorithm 5 that

$$(\forall i \in \{1,...,m\}) \;\; y_{2,i,n} - \tilde{y}_{2,i,n} \to 0 \;\; \text{P-a. s.} \quad \text{and} \quad p_{2,i,n} - \tilde{p}_{2,i,n} \to 0 \;\; \text{P-a. s.} \tag{69}$$

Moreover, we deduce from (ii)(a) that there exists a $u \in \mathcal{H}$ such that

$$u \in A\bar{x} \;\; \text{P-a. s.} \quad \text{and} \quad 0 = u + \sum_{j=1}^{m} L_j^* \bar{v}_j + C\bar{x} \;\; \text{P-a. s.} \tag{70}$$

and that

$$(\forall i \in \{1,...,m\}) \;\; L_i \bar{x} - D_i^{-1}\bar{v}_i \in B_i^{-1}\bar{v}_i \;\; \text{P-a. s.} \tag{71}$$

Additionally, in view of Eq. 65 and [3, Proposition 23.2(ii)] we derive

$$(\forall n \in \mathbb{N}) \;\; \gamma_n^{-1}(x_n - \tilde{p}_{1,n}) - Cx_n - \sum_{j=1}^{m} L_j^* v_{j,n} \in A\tilde{p}_{1,n} \tag{72}$$

while in view of Eq. 66 and [3, Proposition 23.2(ii)] we derive

$$(\forall i \in \{1,...,m\})(\forall n \in \mathbb{N}) \;\; \gamma_n^{-1}(v_{i,n} - \tilde{p}_{2,i,n}) + L_i x_n - D_i^{-1} v_{i,n} \in B_i^{-1}\tilde{p}_{2,i,n}. \tag{73}$$

Now we set

(74)

$$(\forall n \in \mathbb{N}) \begin{cases} \alpha_{1,n} = ||x_n - \tilde{p}_{1,n}|| \big(\rho^{-1}||\tilde{p}_{1,n} - \overline{x}|| + \mu||x_n - \overline{x}|| + \sum_{i=1}^{m} ||L_i||\;||v_{i,n} - \overline{v}_i||\big) \\ \alpha_{2,n} = \sum_{i=1}^{m} (\rho^{-1} + \nu_i)||v_{i,n} - \tilde{p}_{2,i,n}||\;||\tilde{p}_{2,i,n} - \overline{v}_i||. \end{cases}$$

Then we derive from (i), (ii)(c), (ii)(d), Eqs. 68, and 69 that

$$\alpha_{1,n} \to 0 \;\; \text{P-a.s.} \quad \text{and} \quad \alpha_{2,n} \to 0 \;\; \text{P-a.s.} \tag{75}$$

Using the Cauchy-Schwarz inequality, the Lipschitzianity and Monotonicity of $C$, the fact, that $\rho^{-1} \geq \gamma_n^{-1}$ and Eq. 70, we get

$$\begin{aligned}
&(\forall n \in \mathbb{N}) \;\; \alpha_{1,n} + \left\langle x_n - \overline{x}, \sum_{i=1}^{m} L_i^*(\overline{v}_i - v_{i,n}) \right\rangle \\
&\quad \geq ||x_n - \tilde{p}_{1,n}|| \big(\rho^{-1}||\tilde{p}_{1,n} - \overline{x}|| + ||Cx_n - C\overline{x}||\big) \\
&\qquad + \left\langle \tilde{p}_{1,n} - x_n, \sum_{i=1}^{m} L_i^*(\overline{v}_i - v_{i,n}) \right\rangle \\
&\qquad + \left\langle x_n - \overline{x}, \sum_{i=1}^{m} L_i^*(\overline{v}_i - v_{i,n}) \right\rangle \\
&\quad = ||x_n - \tilde{p}_{1,n}|| \big(\rho^{-1}||\tilde{p}_{1,n} - \overline{x}|| + ||Cx_n - C\overline{x}||\big) \\
&\qquad + \left\langle \tilde{p}_{1,n} - \overline{x}, \sum_{i=1}^{m} L_i^*(\overline{v}_i - v_{i,n}) \right\rangle \\
&\quad \geq \left\langle \tilde{p}_{1,n} - \overline{x}, \gamma_n^{-1}(x_n - \tilde{p}_{1,n}) + \sum_{i=1}^{m} L_i^*(\overline{v}_i - v_{i,n}) \right\rangle \\
&\qquad + \langle \tilde{p}_{1,n} - x_n, C\overline{x} - Cx_n \rangle \\
&\quad = \left\langle \tilde{p}_{1,n} - \overline{x}, \gamma_n^{-1}(x_n - \tilde{p}_{1,n}) - \sum_{i=1}^{m} L_i^* v_{i,n} - Cx_n + \sum_{i=1}^{m} L_i^* \overline{v}_i + C\overline{x} \right\rangle \\
&\qquad + \langle \overline{x} - x_n, C\overline{x} - Cx_n \rangle \\
&\quad = \left\langle \tilde{p}_{1,n} - \overline{x}, \gamma_n^{-1}(x_n - \tilde{p}_{1,n}) - \sum_{i=1}^{m} L_i^* v_{i,n} - Cx_n - u \right\rangle \\
&\qquad + \langle \overline{x} - x_n, C\overline{x} - Cx_n \rangle \\
&\quad \geq \left\langle \tilde{p}_{1,n} - \overline{x}, \left( \gamma_n^{-1}(x_n - \tilde{p}_{1,n}) - \sum_{i=1}^{m} L_i^* v_{i,n} - Cx_n \right) - u \right\rangle \;\; \text{P-a.s.}
\end{aligned} \tag{76}$$

Now let $A$ be uniformly monotone at $\overline{x}$. Then, due to Eqs. 70, 72, and 76, there exists an increasing function $\phi_A : [0, +\infty[ \to [0, +\infty]$ that vanishes only at 0 such that

$$(\forall n \in \mathbb{N}) \;\; \alpha_{1,n} + \left\langle x_n - \overline{x}, \sum_{i=1}^{m} L_i^*(\overline{v}_i - v_{i,n}) \right\rangle \geq \phi_A(||\tilde{p}_{1,n} - \overline{x}||) \;\; \text{P-a.s.} \tag{77}$$

On the other hand, Eq. 74, the Lipschitzianity of the operators $(D_i^{-1})_{1\leq i\leq m}$, the fact, that $\rho^{-1}\geq\gamma_n^{-1}$, the Cauchy-Schwarz inequality, Eqs. 71, 73, and the monotonicity of the operators $(B_i^{-1})_{1\leq i\leq m}$ and $(D_i^{-1})_{1\leq i\leq m}$ yield

$$
\begin{aligned}
&(\forall n\in\mathbb{N})\ \ \alpha_{2,n}+\left\langle x_n-\overline{x},\sum_{i=1}^m L_i^*(\tilde{p}_{2,i,n}-\overline{v}_i)\right\rangle\\
&\quad\geq\sum_{i=1}^m\langle\gamma_n^{-1}(v_{i,n}-\tilde{p}_{2,i,n})-D_i^{-1}v_{i,n}+D_i^{-1}\tilde{p}_{2,i,n}+L_i(x_n-\overline{x}),\tilde{p}_{2,i,n}-\overline{v}_i\rangle\\
&\quad=\sum_{i=1}^m\big(\langle\gamma_n^{-1}(v_{i,n}-\tilde{p}_{2,i,n})+L_ix_n-D_i^{-1}v_{i,n}-(L_i\overline{x}-D_i^{-1}\overline{v}_i),\tilde{p}_{2,i,n}-\overline{v}_i\rangle\\
&\qquad+\langle D_i^{-1}\tilde{p}_{2,i,n}-D_i^{-1}\overline{v}_i,\tilde{p}_{2,i,n}-\overline{v}_i\rangle\big)\\
&\quad\geq 0\ \ \text{P-a.s.}
\end{aligned}
\tag{78}
$$

Adding Eqs. 77 and 78, it follows

$$
(\forall n\in\mathbb{N})\ \ \alpha_{1,n}+\alpha_{2,n}+\left\langle x_n-\overline{x},\sum_{i=1}^m L_i^*(\tilde{p}_{2,i,n}-v_{i,n})\right\rangle\geq\phi_A(||\tilde{p}_{1,n}-\overline{x}||)\ \ \text{P-a.s.}
\tag{79}
$$

Then Eq. 75, (ii)(c), (i), Eq. 69 and [3, Lemma 2.41(iii)] imply that $\phi_A(||\tilde{p}_{1,n}-\overline{x}||)\to 0$ P-a.s. and, in turn, that $\tilde{p}_{1,n}\to\overline{x}$ P-a.s. Thus, due to (i) and Eq. 68, we get $x_n\to\overline{x}$ P-a.s. and $p_{1,n}\to\overline{x}$ P-a.s. Likewise, if $C$ is uniformly monotone at $\overline{x}$, there exists an increasing function $\phi_C:[0,+\infty[\to[0,+\infty]$ that vanishes only at 0 such that

$$
(\forall n\in\mathbb{N})\ \ \alpha_{1,n}+\alpha_{2,n}+\left\langle x_n-\overline{x},\sum_{i=1}^m L_i^*(\tilde{p}_{2,i,n}-v_{i,n})\right\rangle\geq\phi_C(||\tilde{x}_n-\overline{x}||)\ \ \text{P-a.s.}
\tag{80}
$$

and we get the same conclusion. That finishes the proof of (ii)(e).

Finally, we prove (ii)(f). Let now $B_i^{-1}$ uniformly monotone at $\overline{v}_i$ for some $i\in\{1,...,m\}$. Then, proceeding as in Eq. 78 , there exists an increasing function $\phi_{B_i}:[0,+\infty[\to[0,+\infty]$ that vanishes only at 0 such that

$$
\begin{aligned}
&(\forall n\in\mathbb{N})\ \ \alpha_{2,n}+\left\langle x_n-\overline{x},\sum_{j=1}^m L_j^*(\tilde{p}_{2,j,n}-\overline{v}_j)\right\rangle\\
&\quad\geq\sum_{j=1}^m\big(\langle\gamma_n^{-1}(v_{j,n}-\tilde{p}_{2,j,n})+L_jx_n-D_j^{-1}v_{j,n}-(L_j\overline{x}-D_j^{-1}\overline{v}_j),\tilde{p}_{2,j,n}-\overline{v}_j\rangle\\
&\qquad+\langle D_j^{-1}\tilde{p}_{2,j,n}-D_j^{-1}\overline{v}_j),\tilde{p}_{2,j,n}-\overline{v}_j\rangle\big)\\
&\quad\geq\sum_{j=1}^m\langle\gamma_n^{-1}(v_{j,n}-\tilde{p}_{2,j,n})+L_jx_n-D_j^{-1}v_{j,n}-(L_j\overline{x}-D_j^{-1}\overline{v}_j),\tilde{p}_{2,j,n}-\overline{v}_j\rangle\\
&\quad\geq\langle\gamma_n^{-1}(v_{i,n}-\tilde{p}_{2,i,n})+L_ix_n-D_i^{-1}v_{i,n}-(L_j\overline{x}-D_j^{-1}\overline{v}_j),\tilde{p}_{2,j,n}-\overline{v}_j\rangle\\
&\quad\geq\phi_{B_i}(||\tilde{p}_{2,i,n}-\overline{v}_i||)\ \ \text{P-a.s.}
\end{aligned}
\tag{81}
$$

Moreover, according to Eqs. 70, 72, 76, and the monotonicity of $A$

$$
(\forall n\in\mathbb{N})\ \ \alpha_{1,n}+\langle x_n-\overline{x},\sum_{j=1}^m L_j^*(\overline{v}_j-v_{j,n})\rangle\geq 0\ \ \text{P-a.s.}
\tag{82}
$$

Thus,

(83)

$$(\forall n \in \mathbb{N}) \;\; \alpha_{1,n} + \alpha_{2,n} + \Big\langle x_n - \bar{x}, \sum_{j=1}^{m} L_j^*(\tilde{p}_{2,j,n} - v_{j,n}) \Big\rangle \geq \phi_{B_i}(||\tilde{p}_{2,i,n} - \bar{v}_i||) \;\; \text{P-a. s.}$$

By proceeding as previously, we deduce that $\tilde{p}_{2,i,n} \to \bar{v}_i$ P-a. s. and thus, via Eq. 69 and (i), that $p_{2,i,n} \to \bar{v}_i$ P-a. s. and $v_{i,n} \to \bar{v}_i$ P-a. s. If we suppose that $D_i^{-1}$ is uniformly monotone at $\bar{v}_i$, by the same arguments we get these conclusions.

□

In the following remark, we comment on the structure of Algorithm 1 and 2.

**Remark 4** (i) *If in Algorithm 2 a stochastic variable $\varepsilon_{i,n} \in \{0,1\}$ is not activated, i.e. $\varepsilon_{i,n} = 0$, $q_{i,n} - y_{i,n}$ need not be computed. But $(y_{1,n}, ..., y_{m,n})$ is needed to compute $\mathbf{z}_n$, thus $\varepsilon_{i,n}$ only effects the sequence $\mathbf{q}_n$. So computational effort per iteration is only saved in the computation of the sequence $\mathbf{q}_n$, the sequences $\mathbf{y}_n$ and $\mathbf{z}_n$ have to be computed conventionally. In the proof above, the operators $\mathbf{M}$ and $\mathbf{Q}$, defined by Eqs. 46 and 47, respectively, have a convenient block structure, the diagonal structure of $\mathbf{M}$ is preserved in $\mathbf{J}_{\gamma_n \mathbf{M}}$, i.e. $\mathbf{J}_{\gamma_n \mathbf{M}}(x_1, ..., x_m) = \prod_{i=1}^{m} (\mathbf{J}_{\gamma_n \mathbf{M}})_i(x_i)$. Therefore, some of the $\varepsilon_{i,n}$ can already operate on the sequence $\mathbf{y}_n$ (instead of $\mathbf{q}_n$) yielding the same numerical result. In turn, instead of computing all components of $\mathbf{y}_n$ and hence $\mathbf{z}_n$, just a few of its components, depending on the probability distribution on the stochastic variables, have to be computed. Thus, the computational effort per iteration decreases further. In Algorithm 1, for all $i \in \{1, ..., m\}$ the stochastic variables $\varepsilon_{2,i,n}$ operate on $y_{2,i,n}$, only for one stochastic variable, namely $\varepsilon_{1,n}$, this is not possible. The reason is that $p_{1,n}$, and hence $y_{1,n}$ are always required to compute $q_{1,n}$ and $q_{2,i,n}$ for all $i \in \{1, ..., m\}$, independent of a concrete realization of the stochastic variables. Moreover, the computation of $q_{1,n}$ in the case of $\varepsilon_{1,n} = 1$ requires $p_{2,i,n}$, and hence $y_{2,i,n}$ for all $i \in \{1, ..., m\}$.*

(ii) *The error-sequences $(a_{\cdot,n})_{n\in\mathbb{N}}$, $(b_{\cdot,n})_{n\in\mathbb{N}}$, and $(c_{\cdot,n})_{n\in\mathbb{N}}$ relax the requirement for exact computation of the operators and their resloventes, respectively, over the course of the iterations. Are the algorithms used in an application, then usually the stochastic variables are all set constantly to zero.*

(iii) *Extra computational effort can be saved in Algorithm 1 at the expense of memory requirement. If $\varepsilon_{1,n-1} = 0$ for a $n \geq 1$ and hence $x_n = x_{n-1}$, some intermediate results from the $(n-1)$-th iteration step computed with $x_{n-1}$ can be stored and again used in the n-th iteration. Analogously, we store and reuse results computed with $v_{i,n-1}$ if $\varepsilon_{2,i,n-1} = 0$. In this sense, Algorithm 1 reads as follows.*

**Algorithm 6:**

1 $store_{Cx} = Cx_0$
2 **for** $i = 1, ..., m$ **do**
3 $\quad store_{L_i x} = L_i x_0$
4 $\quad store_{L_i^* v_i} = L_i^* v_{i,0}$
5 $\quad store_{D_i^{-1} v_i} = D_i^{-1} v_{i,0}$
6 **for** $n = 0, 1, ...$ **do**
7 $\quad y_{1,n} = x_n - \gamma_n (store_{Cx} + \sum_{i=1}^m store_{L_i^* v_i} + a_{1,n})$
8 $\quad p_{1,n} = J_{\gamma_n A}(y_{1,n}) + b_{1,n}$
9 $\quad$ **if** $\varepsilon_{1,n} = 0$ **then**
10 $\quad\quad$ **for** $i = 1, ..., m$ **do**
11 $\quad\quad\quad$ **if** $\varepsilon_{2,i,n} = 1$ **then**
12 $\quad\quad\quad\quad y_{2,i,n} = v_{i,n} + \gamma_n (store_{L_i x} - store_{D_i^{-1} v_i} - a_{2,i,n})$
13 $\quad\quad\quad\quad p_{2,i,n} = J_{\gamma_n B_i^{-1}}(y_{2,i,n}) + b_{2,i,n}$
14 $\quad\quad\quad\quad q_{2,i,n} = p_{2,i,n} + \gamma_n (L_i p_{1,n} - D_i^{-1} p_{2,i,n} - c_{2,i,n})$
15 $\quad\quad\quad\quad v_{i,n+1} = v_{i,n} - y_{2,i,n} + q_{2,i,n}$
16 $\quad\quad\quad\quad store_{L_i^* v_i} = L_i^* v_{i,n+1}$
17 $\quad\quad\quad\quad store_{D_i^{-1} v_i} = D_i^{-1} v_{i,n+1}$
18 $\quad\quad\quad$ **else**
19 $\quad\quad\quad\quad v_{i,n+1} = v_{i,n}$
20 $\quad\quad x_{n+1} = x_n$
21 $\quad$ **else** $[\varepsilon_{1,n} = 1]$
22 $\quad\quad$ **for** $i = 1, ..., m$ **do**
23 $\quad\quad\quad y_{2,i,n} = v_{i,n} + \gamma_n (store_{L_i x} - store_{D_i^{-1} v_i} - a_{2,i,n})$
24 $\quad\quad\quad p_{2,i,n} = J_{\gamma_n B_i^{-1}}(y_{2,i,n}) + b_{2,i,n}$
25 $\quad\quad q_{1,n} = p_{1,n} - \gamma_n (C p_{1,n} + \sum_{i=1}^m L_i^* p_{2,i,n} + c_{1,n})$
26 $\quad\quad x_{n+1} = x_n - y_{1,n} + q_{1,n}$
27 $\quad\quad store_{Cx} = Cx_{n+1}$
28 $\quad\quad$ **for** $i = 1, ..., m$ **do**
29 $\quad\quad\quad store_{L_i x} = L_i x_{n+1}$
30 $\quad\quad\quad$ **if** $\varepsilon_{2,i,n} = 1$ **then**
31 $\quad\quad\quad\quad q_{2,i,n} = p_{2,i,n} + \gamma_n (L_i p_{1,n} - D_i^{-1} p_{2,i,n} - c_{2,i,n})$
32 $\quad\quad\quad\quad v_{i,n+1} = v_{i,n} - y_{2,i,n} + q_{2,i,n}$
33 $\quad\quad\quad\quad store_{L_i^* v_i} = L_i^* v_{i,n+1}$
34 $\quad\quad\quad\quad store_{D_i^{-1} v_i} = D_i^{-1} v_{i,n+1}$
35 $\quad\quad\quad$ **else**
36 $\quad\quad\quad\quad v_{i,n+1} = v_{i,n}$

*Depending on the distribution of the stochastic variables and on the size of m, the following algorithm may provide further benefits with respect to the computational effort. In difference to Algorithm 6, here the sum in line* 8 *is stored as* $store_{\sum_{i=1}^m L_i^* v_i}$ *instead of the single summands* $store_{L_i^* v_i}$*. Of course the algorithm is equivalent to Algorithm 1 and Algorithm 6.*

**Algorithm 7:**

$store_{Cx} = Cx_0$
**for** $i = 1, ..., m$ **do**
    $store_{L_i x} = L_i x_0$
    $store_{L_i^* v_i} = L_i^* v_{i,0}$
    $store_{D_i^{-1} v_i} = D_i^{-1} v_{i,0}$
$store_{\sum_{i=1}^m L_i^* v_i} = \sum_{i=1}^m store_{L_i^* v_i}$
**for** $n = 0, 1, ...$ **do**
    $y_{1,n} = x_n - \gamma_n (store_{Cx} + store_{\sum_{i=1}^m L_i^* v_i} + a_{1,n})$
    $p_{1,n} = J_{\gamma_n A}(y_{1,n}) + b_{1,n}$
    **if** $\varepsilon_{1,n} = 0$ **then**
        **for** $i = 1, ..., m$ **do**
            **if** $\varepsilon_{2,i,n} = 1$ **then**
                $y_{2,i,n} = v_{i,n} + \gamma_n (store_{L_i x} - store_{D_i^{-1} v_i} - a_{2,i,n})$
                $p_{2,i,n} = J_{\gamma_n B_i^{-1}}(y_{2,i,n}) + b_{2,i,n}$
                $q_{2,i,n} = p_{2,i,n} + \gamma_n (L_i p_{1,n} - D_i^{-1} p_{2,i,n} - c_{2,i,n})$
                $v_{i,n+1} = v_{i,n} - y_{2,i,n} + q_{2,i,n}$
                $store_{\sum_{i=1}^m L_i^* v_i} = store_{\sum_{i=1}^m L_i^* v_i} - store_{L_i^* v_i}$
                $store_{L_i^* v_i} = L_i^* v_{i,n+1}$
                $store_{\sum_{i=1}^m L_i^* v_i} = store_{\sum_{i=1}^m L_i^* v_i} + store_{L_i^* v_i}$
                $store_{D_i^{-1} v_i} = D_i^{-1} v_{i,n+1}$
            **else**
                $v_{i,n+1} = v_{i,n}$
        $x_{n+1} = x_n$
    **else** $[\varepsilon_{1,n} = 1]$
        **for** $i = 1, ..., m$ **do**
            $y_{2,i,n} = v_{i,n} + \gamma_n (store_{L_i x} - store_{D_i^{-1} v_i} - a_{2,i,n})$
            $p_{2,i,n} = J_{\gamma_n B_i^{-1}}(y_{2,i,n}) + b_{2,i,n}$
        $q_{1,n} = p_{1,n} - \gamma_n (C p_{1,n} + \sum_{i=1}^m L_i^* p_{2,i,n} + c_{1,n})$
        $x_{n+1} = x_n - y_{1,n} + q_{1,n}$
        $store_{Cx} = Cx_{n+1}$
        **for** $i = 1, ..., m$ **do**
            $store_{L_i x} = L_i x_{n+1}$
            **if** $\varepsilon_{2,i,n} = 1$ **then**
                $q_{2,i,n} = p_{2,i,n} + \gamma_n (L_i p_{1,n} - D_i^{-1} p_{2,i,n} - c_{2,i,n})$
                $v_{i,n+1} = v_{i,n} - y_{2,i,n} + q_{2,i,n}$
                $store_{\sum_{i=1}^m L_i^* v_i} = store_{\sum_{i=1}^m L_i^* v_i} - store_{L_i^* v_i}$
                $store_{L_i^* v_i} = L_i^* v_{i,n+1}$
                $store_{\sum_{i=1}^m L_i^* v_i} = store_{\sum_{i=1}^m L_i^* v_i} + store_{L_i^* v_i}$
                $store_{D_i^{-1} v_i} = D_i^{-1} v_{i,n+1}$
            **else**
                $v_{i,n+1} = v_{i,n}$

A connection to existing work is shown next.

**Remark 5** *If we set the stochastic variables* $(\varepsilon_{1,n}, \varepsilon_{2,1,n}, ..., \varepsilon_{2,m,n}) = (1, ..., 1)$ *for all* $n \in \mathbb{N}$ *and consider* $x_0, v_{i,0}$ *as well as* $a_{1,n}, b_{1,n}, c_{1,n}, a_{2,i,n}, b_{2,i,n}$ *and* $c_{2,i,n}$ *for all* $n \in \mathbb{N}$ *and for all* $i \in \{1, ..., m\}$ *as constant random variables in Theorem 2, then we obtain the statement given in [2, Theorem 3.1] with* $z = 0$ *and* $r_i = 0$ *for all* $i \in \{1, ..., m\}$. *Because in this case, condition (C) in Theorem 2 is given by* $\mathrm{P}[\varepsilon_{1,0} = 1] = 1$ *and* $\mathrm{P}[\varepsilon_{2,i,0} = 1] = 1$ *for all* $i \in \{1, ..., m\}$ *and therefore satisfied, (B) is trivially satisfied and (A) transforms into absolute summability of the error sequences as required in [2, Theorem 3.1].*

# 4 Applications to Convex Optimization

As mentioned above, the computational effort per iteration in Algorithm 1 varies depending on the probability distribution on the stochastic variables. Setting stochastic variables equal to zero reduces the computational effort per iteration, however, as compensation the number of iterations possibly has to be increased. Now we are interested in whether this reduction translates into faster convergence of the sequences generated by the algorithm when it is applied on real world problems. Since the convex subdifferential of a proper, convex and lower semicontinuous function is a maximally monotone operator, Algorithm 1 can be used to solve convex optimization problems. We use the algorithm given in the following theorem to solve a real world problem in order to assess its performance, and to find "good choices" for the probability distribution on the stochastic variables.

We derive the following convex minimization problem by setting $z = 0$ and $r_i = 0$ for all $i \in \{1, ..., m\}$ in [2, Problem 4.1].

## 4.1 Convex Optimization Problem

Let $\mathcal{H}$ be a separable real Hilbert space, let $m$ be a strictly positive integer, let $f \in \Gamma_0(\mathcal{H})$, and let $h : \mathcal{H} \to \mathbb{R}$ be convex and differentiable with a $\mu$-Lipschitzian gradient for some $\mu \in ]0, +\infty[$. For every $i \in \{1, ..., m\}$, let $\mathcal{G}_i$ be a separable real Hilbert space, let $g_i \in \Gamma_0(\mathcal{G}_i)$, let $l_i \in \Gamma_0(\mathcal{G}_i)$ be $1/\nu_i$-strongly convex, for some $\nu_i \in ]0, +\infty[$, and suppose that $L_i : \mathcal{H} \to \mathcal{G}_i$ is a nonzero bounded linear operator. Consider the problem

$$\min_{x \in \mathcal{H}} \quad f(x) + \sum_{i=1}^{m} (g_i \,\square\, l_i)(L_i x) + h(x), \tag{84}$$

and the dual problem

$$\min_{v_1 \in \mathcal{G}_1, ..., v_m \in \mathcal{G}_m} \quad (f^* \,\square\, h^*)\Big( -\sum_{i=1}^{m} L_i^* v_i \Big) + \sum_{i=1}^{m} \big( g_i^*(v_i) + l_i^*(v_i) \big). \tag{85}$$

The following result is an offspring of Theorem 2.

**Theorem 6** *In the Problem given in section 4.1, suppose that*

$$0 \in \operatorname{ran}\Big( \partial f + \sum_{i=1}^{m} L_i^* \big( (\partial g_i \,\square\, \partial l_i)(L_i \,\cdot) \big) + \nabla h \Big). \tag{86}$$

*Let $(a_{1,n})_{n\in\mathbb{N}}$, $(b_{1,n})_{n\in\mathbb{N}}$ and $(c_{1,n})_{n\in\mathbb{N}}$ be sequences of $\mathcal{H}$-valued random variables, and for every $i\in\{1,...,m\}$, let $(a_{2,i,n})_{n\in\mathbb{N}}$, $(b_{2,i,n})_{n\in\mathbb{N}}$ and $(c_{2,i,n})_{n\in\mathbb{N}}$ be sequences of $\mathcal{G}_i$-valued random variables. Futhermore, set*

$$\beta = \max\{\mu, \nu_1, ..., \nu_m\} + \sqrt{\sum_{i=1}^{m} ||L_i||^2}, \tag{87}$$

*let $x_0$ be a $\mathcal{H}$-valued random variable, let $(v_{1,0}, ..., v_{m,0})$ be a $\mathcal{G}_1 \oplus \cdots \oplus \mathcal{G}_m$-valued vector of random variables, let $\rho \in ]0, 1/(\beta+1)[$, let $(\gamma_n)_{n\in\mathbb{N}}$ be in $[\rho, (1-\rho)/\beta]$. Set $\mathbf{D} = \{0,1\}^{m+1} \setminus \{\mathbf{0}\}$, and let for all $n\in\mathbb{N}$ $(\varepsilon_{1,n}, \varepsilon_{2,1,n}, ..., \varepsilon_{2,m,n})$ be identically distributed $\mathbf{D}$-valued random variables, and iterate*

**Algorithm 8:**

**for** $n = 0, 1, ...$ **do**
  $y_{1,n} = x_n - \gamma_n(\nabla h(x_n) + \sum_{i=1}^{m} L_i^* v_{i,n} + a_{1,n})$
  $p_{1,n} = \operatorname{prox}_{\gamma_n f}(y_{1,n}) + b_{1,n}$
  **if** $\varepsilon_{1,n} = 0$ **then**
    **for** $i = 1, ..., m$ **do**
      **if** $\varepsilon_{2,i,n} = 1$ **then**
        $y_{2,i,n} = v_{i,n} + \gamma_n(L_i x_n - \nabla l_i^*(v_{i,n}) - a_{2,i,n})$
        $p_{2,i,n} = \operatorname{prox}_{\gamma_n g_i^*}(y_{2,i,n}) + b_{2,i,n}$
        $q_{2,i,n} = p_{2,i,n} + \gamma_n(L_i p_{1,n} - \nabla l_i^*(p_{2,i,n}) - c_{2,i,n})$
        $v_{i,n+1} = v_{i,n} - y_{2,i,n} + q_{2,i,n}$
      **else**
        $v_{i,n+1} = v_{i,n}$
    $x_{n+1} = x_n$
  **else** $[\varepsilon_{1,n} = 1]$
    **for** $i = 1, ..., m$ **do**
      $y_{2,i,n} = v_{i,n} + \gamma_n(L_i x_n - \nabla l_i^*(v_{i,n}) - a_{2,i,n})$
      $p_{2,i,n} = \operatorname{prox}_{\gamma_n g_i^*}(y_{2,i,n}) + b_{2,i,n}$
    $q_{1,n} = p_{1,n} - \gamma_n(\nabla h(p_{1,n}) + \sum_{i=1}^{m} L_i^* p_{2,i,n} + c_{1,n})$
    $x_{n+1} = x_n - y_{1,n} + q_{1,n}$
    **for** $i = 1, ..., m$ **do**
      **if** $\varepsilon_{2,i,n} = 1$ **then**
        $q_{2,i,n} = p_{2,i,n} + \gamma_n(L_i p_{1,n} - \nabla l_i^*(p_{2,i,n}) - c_{2,i,n})$
        $v_{i,n+1} = v_{i,n} - y_{2,i,n} + q_{2,i,n}$
      **else**
        $v_{i,n+1} = v_{i,n}$

*and for all $n\in\mathbb{N}$ set $\mathscr{E}_n = \sigma\big((\varepsilon_{1,n}, \varepsilon_{2,1,n}, ..., \varepsilon_{2,m,n})\big)$, $\mathcal{X}_{1,n} = \sigma(x_0, ..., x_n)$ and $(\forall i \in \{1, ..., m\})$ $\mathcal{X}_{2,i,n} = \sigma(v_{2,i,0}, ..., v_{2,i,n})$. In addition, assume that the following hold.*

(A) *For all $n\in\mathbb{N}$, let $a_{1,n}$, $b_{1,n}$ and $c_{1,n}$ be $\mathcal{X}_{1,n}$-measurable and $(\forall i\in\{1,...,m\})$ let $a_{2,i,n}$, $b_{2,i,n}$ and $c_{2,i,n}$ be $\mathcal{X}_{2,i,n}$-measurable. Further, let $\sum_{n\in\mathbb{N}} \mathrm{E}(||a_{1,n}|| \,|\, \mathcal{X}_{1,n}) < +\infty$, $\sum_{n\in\mathbb{N}} \mathrm{E}(||b_{1,n}|| \,|\, \mathcal{X}_{1,n}) < +\infty$, $\sum_{n\in\mathbb{N}} \mathrm{E}(||c_{1,n}|| \,|\, \mathcal{X}_{1,n}) < +\infty$, and let for all $i\in\{1,...,m\}$ $\sum_{n\in\mathbb{N}} \mathrm{E}(||a_{2,i,n}|| \,|\, \mathcal{X}_{2,i,n}) < +\infty$, $\sum_{n\in\mathbb{N}} \mathrm{E}(||b_{2,i,n}|| \,|\, \mathcal{X}_{2,i,n}) < +\infty$ and $\sum_{n\in\mathbb{N}} \mathrm{E}(||c_{2,i,n}|| \,|\, \mathcal{X}_{2,i,n}) < +\infty$.*

(B) *For every* $n \in \mathbb{N}$, *let* $\mathscr{E}_n$ *and* $\sigma(\mathcal{X}_{1,n}, \mathcal{X}_{2,1,n}, ..., \mathcal{X}_{2,m,n})$ *be independent.*

(C) *Let* $\mathrm{P}[\varepsilon_{1,0} = 1] > 0$ *and for all* $i \in \{1, ..., m\}$ *let* $\mathrm{P}[\varepsilon_{2,i,0} = 1] > 0$.

*Then the following hold.*

(i) $\sum_{n\in\mathbb{N}} ||x_n - p_{1,n}||^2 < +\infty$ P-a. s. *and*
$(\forall i \in \{1, ..., m\}) \sum_{n\in\mathbb{N}} ||v_{i,n} - p_{2,i,n}||^2 < +\infty$ P-a. s.

(ii) *There exists a solution* $\bar{x}$ *to Eq. 84 and a solution* $(\bar{v}_1, ..., \bar{v}_m)$ *to Eq. 85 such that the following* P-a. s. *hold.*

(a) $-\sum_{i=1}^m L_i^* \bar{v}_i \in \partial f(\bar{x}) + \nabla h(\bar{x})$ *and* $(\forall i \in \{1, ..., m\})$ $L_i \bar{x} \in \partial g_i^* \bar{v}_i + \nabla l_i^*(\bar{v}_i)$.

(b) $x_n \rightharpoonup \bar{x}$ *and* $p_{1,n} \rightharpoonup \bar{x}$.

(c) $(\forall i \in \{1, ..., m\})$ $v_{i,n} \rightharpoonup \bar{v}_i$ *and* $p_{2,i,n} \rightharpoonup \bar{v}_i$.

(d) *Suppose that* $f$ *or* $h$ *is uniformly convex at* $\bar{x}$. *Then* $x_n \to \bar{x}$ *and* $p_{1,n} \to \bar{x}$.

(e) *Suppose that, for some* $i \in \{1, ..., m\}$, $g_i^*$ *or* $l_i^*$ *is uniformly convex at* $\bar{v}_i$. *Then* $v_{i,n} \to \bar{v}_i$ *and* $p_{2,i,n} \to \bar{v}_i$.

*Proof.* (The paradigm to this proof is the proof of [2, Theorem 4.2].) First we define the operators

$$A = \partial f, \quad C = \nabla h, \quad \text{and} \quad (\forall i \in \{1, ..., m\}) \quad B_i = \partial g_i \quad \text{and} \quad D_i = \partial l_i, \tag{88}$$

in order to associate Problem 1.1 with Problem 4.1. It is obviously that Eq. 86 results in Eq. 15 and, using Eqs. 10 and 11, that Algorithm 1 is equal to Algorithm 8. Furthermore, the operators $A$ and $(B_i)_{1\leq i\leq m}$ are maximally monotone by [3, Theorem 20.40], and $C$ is monotone by [3, Theorem 17.10]. In turn, the $1/\nu_i$-strongly convexity of $l_i$ for all $i \in \{1, ..., m\}$ and [3, Corollary 13.33 and Theorem 18.15] imply that $l_i^*$ is Fréchet differentiable on $\mathcal{G}_i$ with a $\nu_i$-Lipschitzian gradient, together with Eq. 10 we get $D_i^{-1} = \nabla L_i^*$. Moreover, we notice that the stochastic requirements (A), (B) and (C) of Theorem 2 are equal to those of Theorem 6. Altogether, Theorem 2 guarantees the existence of a point $\bar{x} \in \mathcal{H}$ such that

$$0 \in \partial f(\bar{x}) + \sum_{i=1}^m L_i^* \big( (\partial g_i \,\square\, \partial l_i)(L_i \bar{x}) \big) + \nabla h(\bar{x}), \tag{89}$$

and of a vector $(\bar{v}_1, ..., \bar{v}_m) \in \mathcal{G}_1 \oplus \cdots \oplus \mathcal{G}_m$ such that

$$(\exists x \in \mathcal{H}) \begin{cases} -\sum_{j=1}^m L_j^* \bar{v}_j \in \partial f(x) + \nabla h(x) \\ (\forall i \in \{1, ..., m\}) \;\; \bar{v}_i \in (\partial g_i \,\square\, \partial l_i)(L_i x), \end{cases} \tag{90}$$

that satisfy (i) and (ii) of Theorem 6. Now it remains to show that $\bar{x}$ solves Eq. 84 and $(\bar{v}_1, ..., \bar{v}_m)$ solves Eq. 85. Since, for every $i \in \{1, ..., m\}$, $\operatorname{dom} l_i^* = \mathcal{G}_i$, by [3, Proposition 24.27] we have

$$(\forall i \in \{1, ..., m\}) \;\; \partial g_i \,\square\, \partial l_i = \partial(g_i \,\square\, l_i). \tag{91}$$

In turn, we derive from [3, Corollary 16.38(iii) and Proposition 17.26(i)] that

$$\partial(f+h) = \partial f + \nabla h. \tag{92}$$

As a result, from Eq. 89 follows that

$$0 \in \partial(f+h)(\bar{x}) + \sum_{i=1}^{m} L_i^* \big(\partial(g_i \,\square\, l_i)\big)(L_i\bar{x}). \tag{93}$$

Moreover, because if follows from Eq. 86 and [3, Proposition 16.5(ii)] that

$$\partial(f+h) + \sum_{i=1}^{m} L_i^* \big(\partial(g_i \,\square\, l_i)\big)(L_i\,\cdot) \subset \partial\Big(f+h+\sum_{i=1}^{m}(g_i \,\square\, l_i)\circ(L_i\,\cdot)\Big), \tag{94}$$

Eq. 93 implies that

$$0 \in \partial\Big(f+h+\sum_{i=1}^{m}(g_i \,\square\, l_i)\circ(L_i\,\cdot)\Big)(\bar{x}). \tag{95}$$

Hence, Fermat's rule [3, Theorem 16.2] claims that $\bar{x}$ solves Eq. 84. Finally, we prove that $(\bar{v}_1,...,\bar{v}_m)$ solves Eq. 85, therefore we notice that the Eqs. 92 and 10 and [3, Proposition 15.2] imply that

$$(\partial f + \nabla h)^{-1} = \big(\partial(f+h)\big)^{-1} = \partial(f+h)^* = \partial(f^* \,\square\, h^*). \tag{96}$$

Similarly, Eq. 91 and [3, Proposition 13.21(i)] yield

$$(\forall i \in \{1,...,m\}) \;\; (\partial g_i \,\square\, \partial l_i)^{-1} = \partial(g_i \,\square\, l_i)^* = \partial(g_i^* + l_i^*). \tag{97}$$

Now we combine the Eqs. 90, 96 and 97 in order to derive

$$(\exists x \in \mathcal{H}) \;\; \begin{cases} x \in \partial(f^* \,\square\, h^*)(-\sum_{j=1}^{m} L_j^*\bar{v}_j) \\ (\forall i \in \{1,...,m\}) \;\; L_i x \in \partial(g_i^* + l_i^*)(\bar{v}_i), \end{cases} \tag{98}$$

and hence

$$(\exists x \in \mathcal{H}) \;\; \begin{cases} -(L_i x)_{1\le i\le m} \in -\Big(\bigtimes_{i=1}^{m} L_i\Big)\Big(\partial(f^* \,\square\, h^*)(-\sum_{j=1}^{m} L_j^*\bar{v}_j)\Big) \\ (L_i x)_{1\le i\le m} \in \bigtimes_{i=1}^{m} \partial(g_i^* + l_i^*)(\bar{v}_i). \end{cases} \tag{99}$$

Let us now introduce the following notation, let $T_i$ be a mapping from $\mathcal{G}_i$ to some set $\mathcal{R}$ for $i \in \{1,...,m\}$, then we define

$$\bigoplus_{i=1}^{m} T_i : \prod_{i=1}^{m} \mathcal{G}_i \to \mathcal{R} : (y_i)_{1\le i\le m} \mapsto \sum_{i=1}^{m} T_i y_i. \tag{100}$$

By this notation and [3, Proposition 16.5(ii) and 16.8] we obtain

$$\begin{aligned} (0,...,0) &\in -\Big(\bigtimes_{i=1}^{m} L_i\Big)\Big(\partial(f^* \,\square\, h^*)\Big(-\sum_{j=1}^{m} L_j^*\bar{v}_j\Big)\Big) + \bigtimes_{i=1}^{m} \partial(g_i^* + l_i^*)(\bar{v}_i) \\ &= -\Big(\bigoplus_{i=1}^{m} L_i^*\Big)^*\Big(\partial(f^* \,\square\, h^*)\Big(-\Big(\bigoplus_{i=1}^{m} L_i^*\Big)(\bar{v}_1,...,\bar{v}_m)\Big)\Big) \\ &\quad + \partial\Big(\bigoplus_{i=1}^{m}(g_i^* + l_i^*)\Big)(\bar{v}_1,...,\bar{v}_m) \\ &\subset \partial\Big((f^* \,\square\, h^*)\Big(-\Big(\bigoplus_{i=1}^{m} L_i^*\Big)\;\cdot\Big) + \bigoplus_{i=1}^{m}(g_i^* + l_i^*)\Big)(\bar{v}_1,...,\bar{v}_m). \end{aligned} \tag{101}$$

In other words, by Fermat's rule, $(\bar{v}_1, ..., \bar{v}_m)$ solves Eq. 85. Finally, the strong convergence assertions in (ii)(d) and (ii)(e) follow from Theorem 2(ii)(e) and (ii)(f) because the uniform convexity of a function $\varphi \in \Gamma_0(\mathcal{H})$ at a point of the domain of $\partial\varphi$ implies the uniform monotonicity of $\partial\varphi$ at that point [11, Section 3.4].

□

In the context of convex optimization, the Remarks 4 and 5 read as follows. The first item allows us to compare Algorithm 8 to its deterministic version, wherein random variables are substituted by elements of the Hilbert spaces. The second one shows how to save extra computational effort.

**Remark 7** (i) *If we set the stochastic variables* $(\varepsilon_{1,n}, \varepsilon_{2,1,n}, ..., \varepsilon_{2,m,n}) = (1, ..., 1)$ *for all* $n \in \mathbb{N}$ *and consider* $x_0, v_{i,0}$ *as well as* $a_{1,n}, b_{1,n}, c_{1,n}, a_{2,i,n}, b_{2,i,n}$ *and* $c_{2,i,n}$ *for all* $n \in \mathbb{N}$ *and for all* $i \in \{1, ..., m\}$ *as constant random variables in Theorem 6, then we obtain the statement given in [2, Theorem 4.2] with* $z = 0$ *and* $r_i = 0$ *for* $i \in \{1, ..., m\}$. *This holds due to the same reasons given in Remark 5.*

(ii) *According to Remark 4 (iii), extra computational effort can be saved in Algorithm 8 at the expense of memory requirement. Depending on the distribution of the stochastic variables and on the size of m, one of the two following algorithms provide more benefits.*

**Algorithm 9:**

$store_{\nabla hx} = \nabla h(x_0)$
**for** $i = 1, ..., m$ **do**
  $store_{L_i x} = L_i x_0$
  $store_{L_i^* v_i} = L_i^* v_{i,0}$
  $store_{\nabla l_i^* v_i} = \nabla l_i^*(v_{i,0})$
**for** $n = 0, 1, ...$ **do**
  $y_{1,n} = x_n - \gamma_n(store_{\nabla hx} + \sum_{i=1}^m store_{L_i^* v_i} + a_{1,n})$
  $p_{1,n} = \text{prox}_{\gamma_n f}(y_{1,n}) + b_{1,n}$
  **if** $\varepsilon_{1,n} = 0$ **then**
    **for** $i = 1, ..., m$ **do**
      **if** $\varepsilon_{2,i,n} = 1$ **then**
        $y_{2,i,n} = v_{i,n} + \gamma_n(store_{L_i x} - store_{\nabla l_i^* v_i} - a_{2,i,n})$
        $p_{2,i,n} = \text{prox}_{\gamma_n g_i^*}(y_{2,i,n}) + b_{2,i,n}$
        $q_{2,i,n} = p_{2,i,n} + \gamma_n(L_i p_{1,n} - \nabla l_i^*(p_{2,i,n}) - c_{2,i,n})$
        $v_{i,n+1} = v_{i,n} - y_{2,i,n} + q_{2,i,n}$
        $store_{L_i^* v_i} = L_i^* v_{i,n+1}$
        $store_{\nabla l_i^* v_i} = \nabla l_i^*(v_{i,n+1})$
      **else**
        $v_{i,n+1} = v_{i,n}$
    $x_{n+1} = x_n$
  **else** $[\varepsilon_{1,n} = 1]$
    **for** $i = 1, ..., m$ **do**
      $y_{2,i,n} = v_{i,n} + \gamma_n(store_{L_i x} - store_{\nabla l_i^* v_i} - a_{2,i,n})$
      $p_{2,i,n} = \text{prox}_{\gamma_n g_i^*}(y_{2,i,n}) + b_{2,i,n}$
    $q_{1,n} = p_{1,n} - \gamma_n(\nabla h(p_{1,n}) + \sum_{i=1}^m L_i^* p_{2,i,n} + c_{1,n})$
    $x_{n+1} = x_n - y_{1,n} + q_{1,n}$
    $store_{\nabla hx} = \nabla h(x_{n+1})$
    **for** $i = 1, ..., m$ **do**
      $store_{L_i x} = L_i x_{n+1}$
      **if** $\varepsilon_{2,i,n} = 1$ **then**
        $q_{2,i,n} = p_{2,i,n} + \gamma_n(L_i p_{1,n} - \nabla l_i^*(p_{2,i,n}) - c_{2,i,n})$
        $v_{i,n+1} = v_{i,n} - y_{2,i,n} + q_{2,i,n}$
        $store_{L_i^* v_i} = L_i^* v_{i,n+1}$
        $store_{\nabla l_i^* v_i} = \nabla l_i^*(v_{i,n+1})$
      **else**
        $v_{i,n+1} = v_{i,n}$

**Algorithm 10:**

$store_{\nabla hx} = \nabla h(x_0)$
**for** $i = 1, ..., m$ **do**
  $store_{L_i x} = L_i x_0$
  $store_{L_i^* v_i} = L_i^* v_{i,0}$
  $store_{\nabla l_i^* v_i} = \nabla l_i^*(v_{i,0})$
$store_{\sum_{i=1}^m L_i^* v_i} = \sum_{i=1}^m store_{L_i^* v_i}$
**for** $n = 0, 1, ...$ **do**
  $y_{1,n} = x_n - \gamma_n(store_{\nabla hx} + store_{\sum_{i=1}^m L_i^* v_i} + a_{1,n})$
  $p_{1,n} = \text{prox}_{\gamma_n f}(y_{1,n}) + b_{1,n}$
  **if** $\varepsilon_{1,n} = 0$ **then**
    **for** $i = 1, ..., m$ **do**
      **if** $\varepsilon_{2,i,n} = 1$ **then**
        $y_{2,i,n} = v_{i,n} + \gamma_n(store_{L_i x} - store_{\nabla l_i^* v_i} - a_{2,i,n})$
        $p_{2,i,n} = \text{prox}_{\gamma_n g_i^*}(y_{2,i,n}) + b_{2,i,n}$
        $q_{2,i,n} = p_{2,i,n} + \gamma_n(L_i p_{1,n} - \nabla l_i^*(p_{2,i,n}) - c_{2,i,n})$
        $v_{i,n+1} = v_{i,n} - y_{2,i,n} + q_{2,i,n}$
        $store_{\sum_{i=1}^m L_i^* v_i} = store_{\sum_{i=1}^m L_i^* v_i} - store_{L_i^* v_i}$
        $store_{L_i^* v_i} = L_i^* v_{i,n+1}$
        $store_{\sum_{i=1}^m L_i^* v_i} = store_{\sum_{i=1}^m L_i^* v_i} + store_{L_i^* v_i}$
        $store_{\nabla l_i^* v_i} = \nabla l_i^*(v_{i,n+1})$
      **else**
        $v_{i,n+1} = v_{i,n}$
    $x_{n+1} = x_n$
  **else** $[\varepsilon_{1,n} = 1]$
    **for** $i = 1, ..., m$ **do**
      $y_{2,i,n} = v_{i,n} + \gamma_n(store_{L_i x} - store_{\nabla l_i^* v_i} - a_{2,i,n})$
      $p_{2,i,n} = \text{prox}_{\gamma_n g_i^*}(y_{2,i,n}) + b_{2,i,n}$
    $q_{1,n} = p_{1,n} - \gamma_n(\nabla h(p_{1,n}) + \sum_{i=1}^m L_i^* p_{2,i,n} + c_{1,n})$
    $x_{n+1} = x_n - y_{1,n} + q_{1,n}$
    $store_{\nabla hx} = \nabla h(x_{n+1})$
    **for** $i = 1, ..., m$ **do**
      $store_{L_i x} = L_i x_{n+1}$
      **if** $\varepsilon_{2,i,n} = 1$ **then**
        $q_{2,i,n} = p_{2,i,n} + \gamma_n(L_i p_{1,n} - \nabla l_i^*(p_{2,i,n}) - c_{2,i,n})$
        $v_{i,n+1} = v_{i,n} - y_{2,i,n} + q_{2,i,n}$
        $store_{\sum_{i=1}^m L_i^* v_i} = store_{\sum_{i=1}^m L_i^* v_i} - store_{L_i^* v_i}$
        $store_{L_i^* v_i} = L_i^* v_{i,n+1}$
        $store_{\sum_{i=1}^m L_i^* v_i} = store_{\sum_{i=1}^m L_i^* v_i} + store_{L_i^* v_i}$
        $store_{\nabla l_i^* v_i} = \nabla l_i^*(v_{i,n+1})$
      **else**
        $v_{i,n+1} = v_{i,n}$

*Of course, Algorithm 8, Algorithm 9 and Algorithm 10 are equivalent. Later on,*

*in Section 4.2.6 we present applications where Algorithm 9 outperforms Algorithm 10, and vice versa.*

## 4.2 Kernel based Machine Learning

The application which we present regards predicting the correct class of images. The approach we use here is referred to as *Support Vector Machines classification* and belongs to the class of *kernel based learning methods*.

We use samples of handwritten gray-style digits of the numbers four and five, taken from a pool consisting of 11339 images provided by the website `http://www.cs.nyu.edu/roweis/data.html`. Each picture has pixel size $28 \times 28$, and each pixel takes values between 0 (black) and 255 (white), see Figure 1. We label the pictures showing a four by $-1$ and the pictures which indicate a five by $+1$, in this case we talk of *supervised learning*, because every sample is labeled. Now we aim to "learn" the *classifier functional* $f$ from the samples, i.e. we want to find a function $f$, which evaluated at an image showing a four is close to $-1$, while $f$ evaluated at a picture of five is close to $+1$. Then we use the *decision function* $\operatorname{sign} f$ to classify digits, where $(\operatorname{sign} f)(x) = \begin{cases} -1 & \text{if } f(x) \leq 0 \\ +1 & \text{otherwise} \end{cases}$. Usually the sample set contains a few outliers, e.g. a digit labeled by $-1$ which nobody would identify as a four. Such an outlier is allowed to be misclassified, to force the decision function to assign outliers to their labeled class would possibly deform it, this effect is called *oversampling*. In order to asses the quality of the decision function, one splits the sample set in a *training data set* and a *test data set*. The former one is used to determine the classifier functional and hence the decision function, which in turn is used to classify the digits of the test data set, the percentage of the misclassified ones is called *misclassification rate*, which rates the quality of the decision function. A similar experimental setup is given in [8, Section 4.3].

All numerical experiments are performed on a 2.3GHz Intel Core i3-2350 machine with 4GB memory. The programs are implemented in MATLAB 7.12.0.

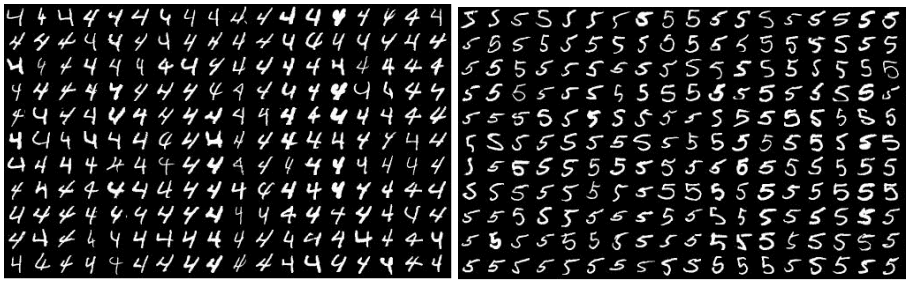

Figure 1 shows a sample of images belonging to the classes $-1$ (four) and $+1$ (five), respectively.

We choose the classifier functional $f$ to be an element of the *Reproducing Kernel Hilbert Space* (RKHS) $\mathcal{H}_\kappa$, which is in our case defined by the symmetric, finitely positive definite and continuous Gaussian kernel function

$$\kappa : \mathbb{R}^l \times \mathbb{R}^l \to \mathbb{R}, \ \kappa(x,y) = \exp\left( - \frac{||x-y||^2}{2\sigma_\kappa^2} \right). \tag{102}$$

Let $\langle \cdot, \cdot \rangle_\kappa$ denote the inner product on $\mathcal{H}_\kappa$. We define the *Gram matrix* $K \in \mathbb{R}^{m \times m}$ with respect to the training data set

$$\mathscr{Z} = \{(X_1, y_1), ..., (X_m, y_m)\} \subset [0,255]^{784(=28\times 28)} \times \{-1, +1\}, \tag{103}$$

namely the symmetric and positive definite matrix with entries $K_{ij} = \kappa(X_i, X_j) \in (0,1]$ for $i, j = 1, ..., m$. The sample images $X_j$ for $j = 1, ..., m$ are in column vector-shape and due to numerical reasons normalized, i.e. divided by $\sqrt{\frac{1}{m}\sum_{i=1}^m ||X_i||^2}$. Without the normalization of the samples, the Gram matrix would be too close to the identity matrix in order to obtain reliable results. To penalize the deviation between the predicted value $f(X)$ and the true value $y \in \{-1, +1\}$, we use the *hinge loss*

$$v_H : \mathbb{R} \times \mathbb{R} \to \mathbb{R} : (x, y) \mapsto \max\{1 - xy, 0\}. \tag{104}$$

To avoid overfitting, we use the *smoothness functional* $\Omega : \mathcal{H}_\kappa \to \mathbb{R}$ to keep the decision function $f$ smooth. Altogether, we get the decision functional as an optimal solution of the *Tikhonov regularization problem*

$$\min_{f \in \mathcal{H}_\kappa} \quad C \sum_{i=1}^m v_H(f(X_i), y_i) + \frac{1}{2}\Omega(f), \tag{105}$$

where $C > 0$ is referred to as *regularization parameter*. It regulates the tradeoff between the sum over the loss functions and the smoothness functional, i.e. the greater $C$ is chosen, the more the decision functional $f$ is forced to classify the training samples correctly.

By the *representer theorem* (cf.[9]), for every $f \in \mathcal{H}_\kappa$ exists a column vector $c = (c_1, ..., c_m)^T \in \mathbb{R}^m$ such that $f$ can be written as $f(\cdot) = \sum_{i=1}^m c_i \kappa(\cdot, X_i)$, and vice versa every such representation for any vector $c \in \mathbb{R}^m$ is a function in the RKHS $\mathcal{H}_\kappa$. In this context we define the smoothness functional $\Omega(f) = ||c||_1$, where $c$ is the vector in the representation of $f$. We have $f(X_j) = \sum_{i=1}^m c_i \kappa(X_j, X_i) = (Kc)_j$, hence the minimization problem 105 is equivalent to the problem

$$\min_{c \in \mathbb{R}^m} \quad C \sum_{i=1}^m v_H((Kc)_i, y_i) + ||c||_1 \tag{106}$$

Let be $\tilde{c}$ a solution to this problem, then the decision function is given by $(\operatorname{sign} \tilde{f})(X) = \operatorname{sign}\left(\sum_{i=1}^m \tilde{c}_i \kappa(X, X_i)\right)$.

In order to solve this problem, we use 2000 samples showing the number four and 2000 samples of fives, thus $m = 4000$. Furthermore, the minimization problem can be represented in context of problem 4.1, i.e. Eq. 106 is equivalent to Eq. 84, if we set

$$\begin{aligned}
&\mathcal{H} = \mathbb{R}^m, \ \mathcal{G}_i = \mathbb{R}, \\
&h : \mathbb{R}^m \to \mathbb{R} : x \mapsto 0, \\
&f : \mathbb{R}^m \to \mathbb{R} : x \mapsto ||x||_1 = \sum_{i=1}^m |x_i|, \\
&(\forall i \in \{1, .., m\}) \ \ l_i : \mathbb{R} \to \overline{\mathbb{R}} : x \mapsto \begin{cases} 0 & \text{if } x = 0 \\ +\infty & \text{otherwise} \end{cases}, \\
&\qquad\qquad g_i : \mathbb{R} \to \mathbb{R} : x \mapsto C v_H(x, y_i), \text{ and} \\
&\qquad\qquad L_i : \mathbb{R}^m \to \mathbb{R} : x \mapsto (Kx)_i = K_{i,:} x,
\end{aligned} \tag{107}$$

where $K_{i,:}$ denotes the $i$-th row of $K$. Therefore, we may use one of the Algorithms 9 and 10 to solve problem 106. All these functions in Eq. 107 satisfy the requirements of Theorem 6 for weak convergence, however not for strong one. Together with Eq. 13 we get $g_i \,\square\, l_i(x) = g_i(x)$. Inserting in the definition of the conjugate function yield $l_i^* \equiv 0$, thus $\nabla l_i^* \equiv 0$. We also have $\nabla h \equiv 0$. Note that $f$ and $g_i$ are not differentiable, thus it is not possible to process them via their gradients. For all $n \geq 0$ we set the error sequences $a_{1,n}, b_{1,n}, c_{1,n}$ and $a_{2,i,n}, b_{2,i,n}, c_{2,i,n}$ for $i \in \{1, ...m, \}$ equal to 0, hence condition (A) of Theorem 6 is satisfied (cf. Remark 4 (ii)). Since $g_i$ is real-valued for all $i$, [2, Proposition 4.3, (ii)] applies, thus condition 86 is satisfied.

Usually the optimization problem 106 is solved using several values for the regularization parameter $C$ and the Gaussian kernel parameter $\sigma_\kappa$ to find good choices for them, here good means for example minimizing the misclassification rate. Although, we are more concerned about the convergence behavior of the algorithm when solving the optimization problem, i.e. finding good choices for $\gamma_n$ and the stochastic variables $\varepsilon_{\cdot,n}$, that is why we fix $C = 1$ and $\sigma_\kappa = 1$ for all numerical experiments in this section.

### 4.2.1 Computation of the Operators needed in the Algorithms

In order to make the algorithm run, we have to calculate the proximal points for $\gamma_n f$ and $\gamma_n g_i^*$ and the operator $L_i^*$, we start with the computation of $\text{prox}_{\gamma_n g_i^*}$. We know from the extended Moreau decomposition formula [10, Theorem 6.45] that

$$\text{prox}_{\gamma_n g_i^*}(x) = x - \gamma_n \,\text{prox}_{\frac{1}{\gamma_n} g_i}\left(\frac{1}{\gamma_n} x\right). \tag{108}$$

Therefore, we compute

$$\text{prox}_{\frac{1}{\gamma_n} g_i}(x) = \arg\min_u \left\{ \frac{g_i(u)}{\gamma_n} - \frac{||x-u||^2}{2} \right\} = \arg\min_u \left\{ \frac{C\max\{1-uy_i, 0\}}{\gamma_n} - \frac{(x-u)^2}{2} \right\}. \tag{109}$$

In order to solve this strongly convex minimization problem, we make a case distinction and set the derivative of $G(u) = \frac{C\max\{1-uy_i,0\}}{\gamma_n} - \frac{(x-u)^2}{2}$ equal to 0.

$$\begin{aligned}
&\underline{\text{Case } 1-uy_i \geq 0}: G'(u) = 0 \iff -\frac{Cy_i}{\gamma_n} + u - x = 0 \iff u = x + \frac{Cy_i}{\gamma_n}, \\
&\qquad 1-uy_i \geq 0 \Rightarrow 1 - uy_i = 1 - xy_i - \frac{Cy_i^2}{\gamma_n} \iff 1 - \frac{C}{\gamma_n} \geq xy_i, \text{ since } y_i^2 = 1. \\
&\underline{\text{Case } 1-uy_i < 0}: G'(u) = 0 \iff u = x, \\
&\qquad 1-uy_i < 0 \Rightarrow 1 < xy_i. \\
&\underline{\text{Case } 1 - \frac{C}{\gamma_n} < xy_i \leq 1 : u = y_i.}
\end{aligned} \tag{110}$$

Thus, we obtain

$$\text{prox}_{\frac{1}{\gamma_n} g_i}(x) = \begin{cases} x & \text{if } xy_i > 1 \\ y_i & \text{if } 1 - \frac{C}{\gamma_n} < xy_i \leq 1. \\ x + \frac{Cy_i}{\gamma_n} & \text{if } 1 - \frac{C}{\gamma_n} \geq xy_i \end{cases} \tag{111}$$

Hence, combining Eq. 108 and 111 yield

$$\begin{aligned}\text{prox}_{\gamma_n g_i^*}(x) &= x-\gamma_n\begin{cases}\frac{x}{\gamma_n} & \text{if } \frac{xy_i}{\gamma_n}>1\\ y_i & \text{if } 1-\frac{C}{\gamma_n}<\frac{xy_i}{\gamma_n}\le 1\\ \frac{x+Cy_i}{\gamma_n} & \text{if } 1-\frac{C}{\gamma_n}\ge\frac{xy_i}{\gamma_n}\end{cases}\\ &= \begin{cases}0 & \text{if } \gamma_n - xy_i<0\\ x-\gamma_n y_i & \text{if } 0\le\gamma_n-xy_i<C\\ -Cy_i & \text{if } C\le\gamma_n-xy_i\end{cases} = \mathscr{P}_{-y_i[0,C]}(x-\gamma_n y_i),\end{aligned} \tag{112}$$

where $\mathscr{P}$ denotes the projection operator. Implementing a function in a programming language, we want to avoid case distinctions since they are time consuming and therefore slow down the algorithm essentially if the function is called often. In this sense, we write $\text{prox}_{\gamma_n g_i^*}$ as follows.

$$\begin{aligned}\text{prox}_{\gamma_n g_i^*}(x) &= \begin{cases}\mathscr{P}_{[0,C]}(x+\gamma_n) & \text{if } -y_i=1\\ \mathscr{P}_{[-C,0]}(x-\gamma_n) & \text{if } -y_i=-1\end{cases}\\ &= \begin{cases}\min\big(\max(x+\gamma_n,0),C\big) & \text{if } y_i=-1\\ \max\big(\min(x-\gamma_n,0),-C\big) & \text{if } y_i=1\end{cases}\\ &= \begin{cases}y_i\max\big(y_i^2\min(y_ix-\gamma_n,0),-C\big) & \text{if } y_i=-1\\ y_i\max\big(\min(y_ix-\gamma_n,0),-C\big) & \text{if } y_i=1\end{cases}\\ &= y_i\max\big(\min(y_ix-\gamma_n,0),-C\big).\end{aligned} \tag{113}$$

We used that $C>0$ and in the second to the last transformation step the fact that $\min(x,y)=-\max(-x,-y)$ and vise versa.

A similar calculation as above yield the proximal point operator

$$\text{prox}_{\gamma_n||\cdot||_1}(x) = \left(\begin{cases}x_1-\gamma_n & \text{if } x_1>\gamma_n\\ x_1+\gamma_n & \text{if } x_1<-\gamma_n\\ 0 & \text{otherwise}\end{cases},...,\begin{cases}x_m-\gamma_n & \text{if } x_m>\gamma_n\\ x_m+\gamma_n & \text{if } x_m<-\gamma_n\\ 0 & \text{otherwise}\end{cases}\right). \tag{114}$$

Next we compute $L_i^*:\mathbb{R}\to\mathbb{R}^m$, inserting in its definition yields

$$\begin{aligned}&(\forall x\in\mathbb{R}^m)(\forall y\in\mathbb{R}):\\ &\langle x,L_i^Ty\rangle=\langle L_ix,y\rangle=\langle(Kx)_i,y\rangle=(Kx)_iy=x^T((K_{i,:})^Ty)=\langle x,(K_{i,:})^Ty\rangle,\end{aligned} \tag{115}$$

and since $K$ is symmetric and real, we obtain $L_i^*(y)=L_i^Ty=K_{:,i}\,y$.

Finally, we have to choose values for the sequence $(\gamma_n)_{0\le n<\infty}\subset\,]0,1/\beta[$ and a probability distribution on the stochastic variables in order to start the numerical experiment. Since $\nabla h$ is $\mu$-Lipschitzian for any $\mu>0$, and since $l_i$ is $1/\nu_i$-strongly convex for any $\nu_i>0$ for $i\in\{1,...,m\}$, we set according to Eq. 87, $\beta>\sqrt{\sum_{i=1}^m||L_i||^2}=\sqrt{\sum_{i=1}^m(\max_j|K_{ij}|)^2}=\sqrt{m}=\sqrt{4000}$, hence $\gamma_n<(4000)^{-1/2}$. The starting points $x_0$ and $(v_{1,0},...,v_{m,0})$ are chosen randomly.

### 4.2.2 Testing several Probability Distributions

Let us now define different probability distributions on the stochastic variables $\varepsilon_{\cdot,n}\in\{0,1\}$. Therefore we permute the vector $(1,2,...,m)\mapsto(\pi_1,\pi_2,...,\pi_m)$, $\pi_i\in\{1,...,m\}$ for $1\le i\le m$ and $\pi_i\neq\pi_j$ for $1\le i<j\le m$, randomly (with uniform distribution),

| **Computations** | $K \cdot randn(m,1)$ | $K_{400} = K(1:400,:)$ | $K_{400} \cdot randn(m,1)$ |
|---|---|---|---|
| **Time** (seconds) | 0.015 | 0.011 | 0.002 |

Table 1 shows the average times needed for performing the respective computation 10 times in MATLAB, $randn(m,1)$ is a $m$-dimensional column vector whose components are standard normal distributed, $K$ denotes the $10^4$ by $10^4$ Gram matrix and $K(1:400,:)$ is the MATLAB-notation for the sub-matrix consisting of the first 400 rows of $K$.

$m = 4000$ in our example, then we divide it by a fixed batch size $bs \leq m$ into $len$ batches

$$\begin{aligned} &MB(0) = \{\pi_1, ..., \pi_{bs}\}, \\ &MB(1) = \{\pi_{bs+1}, ..., \pi_{2\cdot bs}\}, \\ &\vdots \\ &MB(len-2) = \{\pi_{(len-2)\cdot bs+1}, ..., \pi_{(len-1)\cdot bs}\}, \\ &MB(len-1) = \{\pi_{(len-1)\cdot bs+1}, ..., \pi_m\}, \end{aligned} \tag{116}$$

where $len = \lceil m/bs \rceil$ and $\lceil a \rceil$ rounds to the nearest integer greater than or equal to $a \in \mathbb{R}$. Now we set

$$\begin{aligned} \varepsilon_{2,i,n} = 1 \ \text{ if } \ i \in MB\big((n \mod len)\big) \text{ for } i \in \{1, ..., m\} \\ \varepsilon_{1,n} = 1 \ \text{ if } \ (n \mod len) = len - 1, \end{aligned} \tag{117}$$

where $(n \mod len)$ denotes the remainder of the integer division $\frac{n}{len}$. Each after $len$ iterations every stochastic variable has been activated, i.e. equal to 1, with equal frequency. Note that condition (C) of Theorem 6 is satisfied, however, condition (B) is not. The reason why we fix the distribution of the stochastic variables in advance and thus violate condition (B) lies in the encoding of the algorithm. Every iteration, the program has to access to those lines of the Gram matrix $K$ which have activated indices, i.e. every line $i$ of $K$ with $\varepsilon_{2,i,n} = 1$ for $1 \leq i \leq m$. It is coded in the lines 16 and 33 in Algorithm 9 and in the lines 18 and 37 in Algorithm 10, respectively, wherein $L_i^* v_{i,n+1} = K(i,:)v_{i,n+1}$ has to be computed for all $i$ with $\varepsilon_{2,i,n} = 1$ ($K(:,i)$ is the MATLAB-notation for the $i$-th column of $K$). This accessing-operation requires proportionally much time in comparison to a computation (e.g. multiplication) with the remaining part of the matrix, see Table 1. Using the distribution scheme of the stochastic variables defined by the Eqs. 116 and 117, it is sufficient to divide the Gram matrix $K$ just once for all iterations into the blocks $K(MB(0),:), ..., K(MB(len-1),:)$ (MATLAB notation), store them and use the same block every $len$-th iteration. If another distribution scheme satisfies condition (B) of Theorem 6, the vector $[\varepsilon_{2,1,n}, ..., \varepsilon_{2,m,n}]$ has to be randomly updated every iteration as well as the activated lines $K([\varepsilon_{2,1,n}, ..., \varepsilon_{2,m,n}] == 1,:)$ (MATLAB notation) of $K$ have to be accessed every iteration. Therefore such a distribution scheme slows down the algorithms essentially and could destroy the (possible) benefits obtained by saving computational effort by using stochastic variables.

We perform the algorithms using the following concrete distributions.

(i) Type *Det*, $(\varepsilon_{1,n}, \varepsilon_{2,1,n}, ..., \varepsilon_{2,m,n}) = (1, ..., 1)$ for all required $n \geq 0$.

(ii) Type *Stoch* 2, $bs = 2000$ $(len = 2)$.

(iii) Type *Stoch* 10, $bs = 400$ $(len = 10)$.

(iv) Type *Stoch* 50, $bs = 80$ $(len = 50)$.

Due to Remark 7 (i), the algorithms performed by using Type *Det* can be seen as deterministic. Note that type *Stoch* 2 saves almost half of the computational effort per iteration, type *Stoch* 10 almost 90% and type *Stoch* 50 almost 98%. To perform the types *Det* and *Stoch* 2 we use Algorithm 9 and for the types *Stoch* 10 and *Stoch* 50 Algorithm 10, that are the best choices for each type. We give the explanation therefore in Section 4.2.6.

### 4.2.3 Performance of the Algorithms within 5 seconds

For each type we test several choices for $\gamma_n$, all chosen constant for all required $n$. All algorithms are terminated after 5 seconds, the outcome is shown in Figure 2. It displays the courses of the objective function values of problem 106 along the time in a separate window for each type, the objective values are computed with $c = x_n$ and $c = (v_{1,n}, ..., v_{m,n})$, notice that both sequences converge to the minimizer by Theorem 6. The time values shown here and in the following are total computation times for performing all computations needed for each algorithm. We observe that the objective function values computed with $c = (v_{1,n}, ..., v_{m,n})$ are greater by an order of magnitude than those computed with $c = x_{1,n}$, thus we restrict us in the remaining part of this section to analyze the courses of the objective function values computed with $c = x_{1,n}$. In order to present all figures in this section clearly, not every pair of objective value and corresponding time taken after each iteration is indicated. The choice $\gamma_n = 4 \cdot 10^{-4}$ proves do be the best one for the type *Det*, $\gamma_n = 3.2 \cdot 10^{-4}$ for type *Stoch* 2, $\gamma_n = 4.2 \cdot 10^{-4}$ for type *Stoch* 10 while the choice $\gamma_n = 4.4 \cdot 10^{-4}$ is the best one for the type *Stoch* 50 within a runtime of 5 seconds.

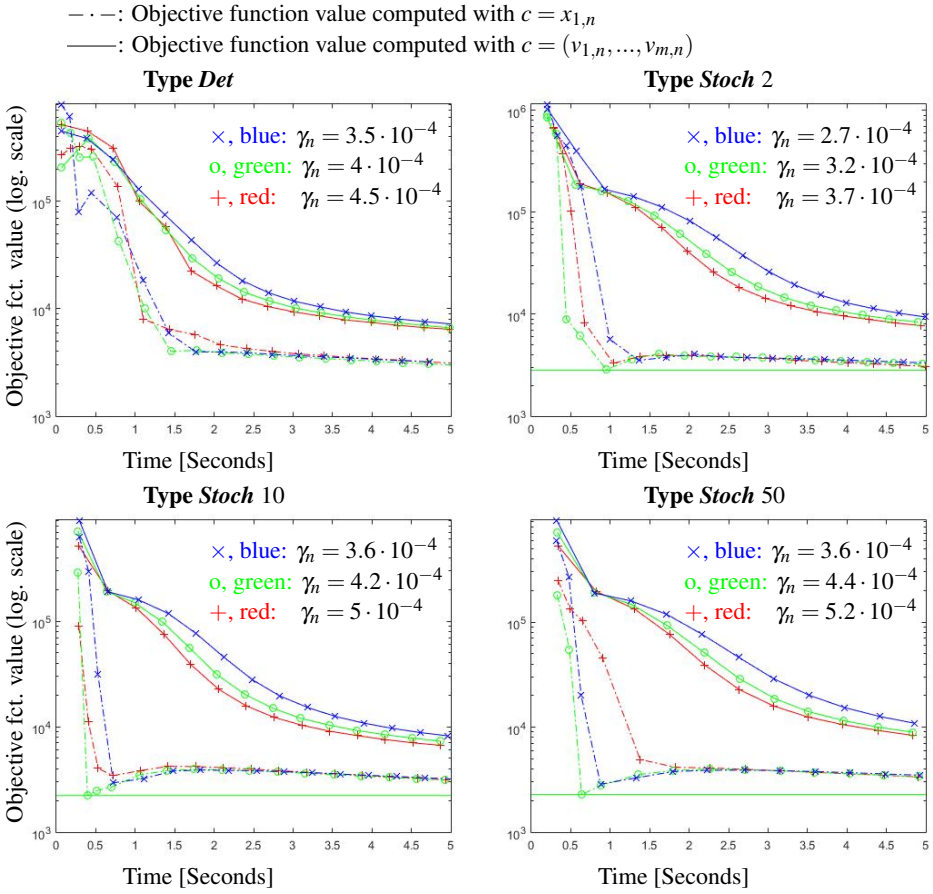


Figure 2 shows the courses of the objective function values for different values of $\gamma_n$, each distribution type is presented in a separate window.

Let us now compare the courses of the objective function values of the four different types of probability distribution, each performed with its best value for $\gamma_n$, the result is shown in Figure 3. The types *Stoch* 2, *Stoch* 10 and *Stoch* 50 have essentially the same convergence behavior, their objective function values decrease rapidly within the first second, then raise a bit but decrease slightly again afterwards. In turn, the course of type *Det* has a shaky start but then decreases continuously, and after 1.5 seconds it has the same course as the other types. The lowest objective function value within 5 seconds is attained by type *Stoch* 10 after 0.4 seconds. The times and numbers of the iterations after each type reached its best objective value and the objective values itself are indicated in Table 2. Therein, the lowest objective function value and the lowest time after a minimum is reached are indicated in bold. The misclassification rate with respect to the training set and the test set for each type, computed respectively with the $x_{1,n}$ corresponding to the lowest objective value is also reported in Table 2. We observe that for three types the misclassification rate with respect to the test set is smaller than those with respect to the training set, that is unusual. Further, we see that a smaller objective value need not necessarily lead to a smaller misclassification rate. Figure 3 also shows us that the type *Det* terminates its first iteration essentially faster than the stochastic types, this is because the Gram matrix has to be split into blocks just once at

| The smallest objective function value of each type computed with $c = x_{1,n}$ was reached at | | | | | Misclassification rate with respect to the | |
|---|---|---|---|---|---|---|
| **Type** | $\gamma_n$ | objective fct. value | Iteration | Time (s) | Training samples | Test samples |
| *Det* | $4 \cdot 10^{-4}$ | 2956 | 92 | 5.01 | 3.33% | 3.08% |
| *Stoch* 2 | $3.2 \cdot 10^{-4}$ | 2843 | 24 | 0.82 | 6.47% | 5.94% |
| *Stoch* 10 | $4.2 \cdot 10^{-4}$ | **2242** | 30 | **0.4** | 7.98% | 8.18% |
| *Stoch* 50 | $4.4 \cdot 10^{-4}$ | 2405 | 250 | 0.65 | 6.5% | 5.98% |

Table 2 shows the statistics to Figure 3, when the smallest objective function value of each type was reached, and the corresponding misclassification rates.

the beginning of the algorithm as described in Section 4.2.2 and performed by the code *MatrixBlocking* presented in Section 4.2.6.

**Comparison of the objective values of Type *Det* / Type *Stoch* 2, 10, 50 within 5 sec.**

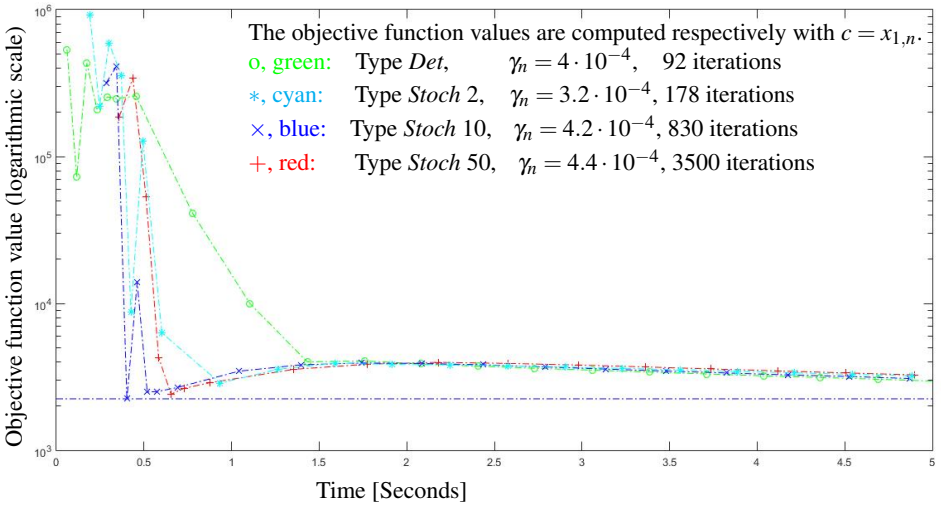


Figure 3 shows the respective course of the objective function values within 5 seconds of the types *Det*, *Stoch* 2, 10 and 50, each performed with its best choice of $\gamma_n$.

### 4.2.4 Performance of the Algorithms within 60 seconds

At the end let us test the performance of the algorithms using the four different types of probability distribution within one minute, the result is shown in Figure 4. Numerical experiments have shown that the choice $\gamma_n = 4.4 \cdot 10^{-4}$ is the best for all types. We observe again that the objective function values computed with $c = (v_{1,n}, ..., v_{m,n})$ are not competitive with those computed with $c = x_{1,n}$. The types *Stoch* 10 and 50 cannot keep up with the other two types. After 5 seconds the types *Det* and *Stoch* 2 have basically the same course of the objective function values computed with $c = x_{1,n}$, which leads to the lowest objective function value after one minute. Every type attained its lowest objective value after the last iteration. The statistics to Figure 4 are indicated in Table 3. Therein, the lowest objective function value is indicated in bold. The misclassification rate with respect to the training set and the test set for each type, computed

respectively with the $x_{1,n}$ corresponding to the lowest objective value is also reported in Table 3. We observe that the misclassification rates obtained here are much better than those obtained after a runtime of 5 seconds. For example, the misclassification rate 1.18% with respect to the test set, which consists of 1784 samples, provided by type *Stoch* 10 means that just 21 samples are wrong classified. Furthermore, Figure 5 shows the best and worst classified images showing four and five computed with $x_{1,9870}$ corresponding to the lowest objective value of type *Stoch* 10. Here best means that the value of the classifier functional is closest to $-1$ and $+1$, respectively, and worst means it is most positive and negative, respectively.

The number of iterations each type did after 5 and 60 seconds is written in the Figures 3 and 4, respectively. Performing the algorithm using type *Stoch 2*, half of the stochastic variables are deactivated and thus a bit more than half of the computational effort per iteration of type *Det* has to be done. This "a bit more" comes from the fact, that in every iteration certain computations (fixed computational work) always have to be performed, independent from the distribution of the stochastic variables, e.g. line 7 and 8 in Algorithm 9 and line 8 and 9 in Algorithm 10. Hence, we expect a bit less than two times the number of iteration done by type *Det*. The numerical experiments presented in the Figures 3 and 4 show that type *Stoch* 2 performs circa 1.96 times as much iterations as the deterministic algorithm. The types *Stoch* 10 and 50 do in average 9.02 and 38.08, respectively, times as much iterations as type *Det*. This numbers conform that the quotient of reduced computational work to fixed computational work per iteration decreases by decreasing the amount of activated stochastic variables.

**Comparison of the objective values of Type *Det* / Type *Stoch* 2, 10, 50 within 1 min.**

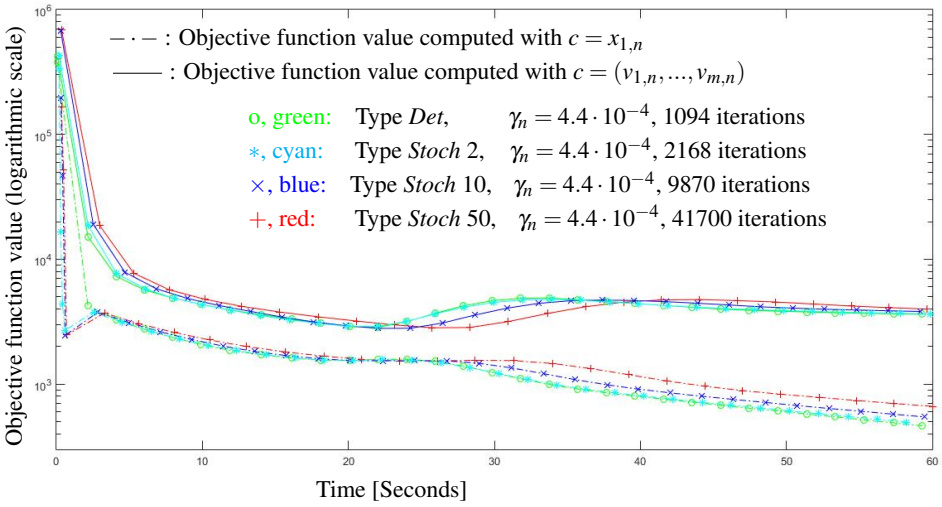


Figure 4 shows the respective course of the objective function values within one minute of the types *Det*, *Stoch* 2, 10 and 50, each performed with $\gamma_n = 4.4 \cdot 10^{-4}$.

| | | | Misclassification rate w. r. t. the | |
|---|---|---|---|---|
| **Type** | $\gamma_n$ | objective fct. value | Training samples | Test samples |
| *Det* | $4.4 \cdot 10^{-4}$ | **455** | 1.15% | 1.35% |
| *Stoch* 2 | $4.4 \cdot 10^{-4}$ | 475 | 1.23% | 1.35% |
| *Stoch* 10 | $4.4 \cdot 10^{-4}$ | 538 | 1.08% | 1.18% |
| *Stoch* 50 | $4.4 \cdot 10^{-4}$ | 663 | 1.18% | 1.23% |

Table 3 shows the statistics to Figure 4, the smallest objective function value of each type and the corresponding misclassification rates.

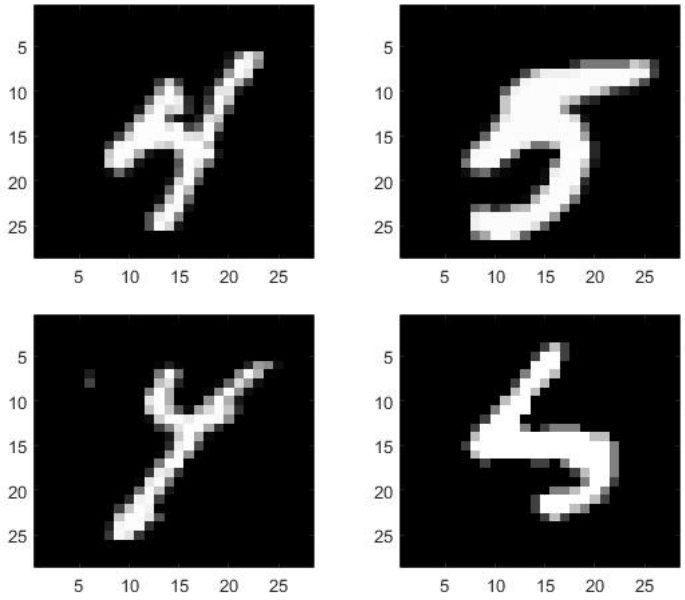


Figure 5 shows in the left column two fours and in the right column two fives, those in the first line are correct classified and those in the second line are wrong classified with respect to the best value obtained by type *Stoch* 10.

### 4.2.5 Conclusion

Finally, we conclude that performing the stochastic algorithm instead of the deterministic one does not improve the results for this application. Within a run-time of one second the stochastic Algorithm 10 using distribution type *Stoch* 10 or 50 provides essentially lower objective function values as the deterministic one, but the performance of the Support Vector Machine corresponding to these objective function values, i.e. its misclassification rate is bad. After a longer runtime, these types cannot keep up anymore. However, the stochastic Algorithm 9 in combination with distribution type *Stoch* 2 provides after a run-time of 10 seconds the same course of the objective values as the deterministic algorithm, which lead to a good performance of the Support Vector Machine. Unfortunately, we could not find a probability distribution to the stochastic variables, which used in a stochastic algorithm outperforms the deterministic one after a longer runtime.

### 4.2.6 Implementation of the Algorithms in MATLAB

Here we present the MATLAB codes to the main routines and its sub-functions. If a code line is too long for the width of the page, we make a word wrap and indent the new line. Some explanations to the code are given in the following as well.

The following code is based on Algorithm 9 and we used it to perform the numerical experiments for the types *Stoch* 2 ($batchSize = 2000$) and *Det* ($batchSize = 4000$).

```
function [ OFV_x, OFV_v, TimePerIt, Min_x, Min_v ]
    = StochasticAlgV1( x, v, K, y, gamma, MaxTime, C, batchSize )
% INPUT PARAMETERS:
% x, v ... starting points.
% K ... Gram matrix corresponding to the training data set.
% y ... labels (-1,+1) corresponding to the training data set.
% gamma ... step size.
% maxTime ... time after which the algorithm stops.
% C ... regularization parameter.
% batchSize ... number of stochastic variables which are
%   set to 1 per iteration.

% OUTPUT PARAMETERS:
% OFV_x ... list of obj. fct. values
%   computed with x, respecticely taken after a iteration.
% OFV_v ... list of obj. fct. values
%   computed with v, respecticely taken after a iteration.
% TimePerIt ... list of times,
%   respecticely taken after a iteration.
% Min_x ... list of x, respecticely taken after a iteration.
% Min_v ... list of v, respecticely taken after a iteration.
tic
m=size(K,1);
%------Initialisation-------------
OFV_x=[];
OFV_v=[];
TimePerIt=[];
Min_x=[];
Min_v=[];
totalTime=0;
Kx=K*x;
[Block, epsilon]=MatrixBlocking( K, batchSize );
KvElement=zeros(m,length(Block));
for j=1:length(Block)
    KvElement(:,j)=Block{j}'*v(epsilon{j});
end
%---------------------------------
while totalTime < MaxTime
    for j=1:length(Block)
        Kv=sum(KvElement,2);
        y1=x-gamma*Kv; % K is symmetric, i.e. K=K'.
        p1=proxNorm1( y1, gamma );
        if j ~= length(Block)
            y2=v(epsilon{j})+gamma*(Kx(epsilon{j}));
            p2=proxConjugateHingeLoss( y2, y(epsilon{j}), gamma, C );
            q2=p2+gamma*(Block{j}*p1);
            v(epsilon{j})=v(epsilon{j})-y2+q2;
        else
            y2=v+gamma*(Kx);
            p2=proxConjugateHingeLoss( y2, y, gamma, C );
            q1=p1-gamma*(K*p2); % K is symmetric, i.e. K=K'.
            x=x-y1+q1;
```

```
53              Kx=K*x;
54              q2=p2(epsilon{j})+gamma*(Block{j}*p1);
55              v(epsilon{j})=v(epsilon{j})-y2(epsilon{j})+q2;
56          end
57          KvElement(:,j)=Block{j}'*v(epsilon{j});
58      end
59      timePerIteration=toc;
60      %------Computing the Output-------
61      Min_x=[Min_x,x];
62      Min_v=[Min_v,v];
63      valx=C*sum(max(ones(m,1)-(K*x).*y,0))+norm(x,1);
64      valv=C*sum(max(ones(m,1)-(K*v).*y,0))+norm(v,1);
65      OFV_x=[OFV_x,valx];
66      OFV_v=[OFV_v,valv];
67      TimePerIt=[TimePerIt,timePerIteration];
68      %---------------------------------
69      totalTime=totalTime+timePerIteration;
70      tic
71  end
72  end
```

The next code is based on Algorithm 10 and we used it to perform the numerical experiments for the types *Stoch* 10 ($batchSize = 400$) and *Stoch* 50 ($batchSize = 80$). If we perform the above code *StochasticAlgV1* for the types *Stoch* 10 and *Stoch* 50, in line 40, 9 and 49, respectively, $m$-dimensional vector additions have do be done every iteration. However, in the subsequent code *StochasticAlgV2* just one $m$-dimensional vector addition and one $m$-dimensional vector subtraction have to be done in the lines 57 and 59 yielding the same result. Due to the same arguments the program *StochasticAlgV1* is the better choice for type *Stoch* 2, since just one addition has to be done in line 40.

```
1   function [ OFV_x, OFV_v, TimePerIt, Min_x, Min_v ]
2       = StochasticAlgV2( x, v, K, y, gamma, MaxTime, C, batchSize )
3   % INPUT PARAMETERS:
4   % x, v ... starting points.
5   % K ... Gram matrix corresponding to the training data set.
6   % y ... labels (-1,+1) corresponding to the training data set.
7   % gamma ... step size.
8   % maxTime ... time after which the algorithm stops.
9   % C ... regularization parameter.
10  % batchSize ... number of stochastic variables which are
11  % set to 1 per iteration.
12
13  % OUTPUT PARAMETERS:
14  % OFV_x ... list of obj. fct. values
15  %   computed with x, respecticely taken after a iteration.
16  % OFV_v ... list of obj. fct. values
17  %   computed with v, respecticely taken after a iteration.
18  % TimePerIt ... list of times,
19  %   respecticely taken after a iteration.
20  % Min_x ... list of x, respecticely taken after a iteration.
21  % Min_v ... list of v, respecticely taken after a iteration.
22  tic
23  m=size(K,1);
24  %------Initialisation--------------
25  OFV_x=[];
26  OFV_v=[];
27  TimePerIt=[];
28  Min_x=[];
```

```
29  Min_v=[];
30  totalTime=0;
31  Kx=K*x;
32  [Block, epsilon]=MatrixBlocking( K, batchSize );
33  KvElement=zeros(m,length(Block));
34  for j=1:length(Block)
35      KvElement(:,j)=Block{j}'*v(epsilon{j});
36  end
37  Kv=sum(KvElement,2);
38  %---------------------------------
39  while totalTime < MaxTime
40      for j=1:length(Block)
41          y1=x-gamma*Kv; % K is symmetric, i.e. K=K'.
42          p1=proxNorm1( y1, gamma );
43          if j ~= length(Block)
44              y2=v(epsilon{j})+gamma*(Kx(epsilon{j}));
45              p2=proxConjugateHingeLoss( y2, y(epsilon{j}), gamma, C );
46              q2=p2+gamma*(Block{j}*p1);
47              v(epsilon{j})=v(epsilon{j})-y2+q2;
48          else
49              y2=v+gamma*(Kx);
50              p2=proxConjugateHingeLoss( y2, y, gamma, C );
51              q1=p1-gamma*(K*p2); % K is symmetric, i.e. K=K'.
52              x=x-y1+q1;
53              Kx=K*x;
54              q2=p2(epsilon{j})+gamma*(Block{j}*p1);
55              v(epsilon{j})=v(epsilon{j})-y2(epsilon{j})+q2;
56          end
57          Kv=Kv-KvElement(:,j);
58          KvElement(:,j)=Block{j}'*v(epsilon{j});
59          Kv=Kv+KvElement(:,j);
60      end
61      timePerIteration=toc;
62      %------Computing the Output-------
63      Min_x=[Min_x,x];
64      Min_v=[Min_v,v];
65      valx=C*sum(max(ones(m,1)-(K*x).*y,0))+norm(x,1);
66      valv=C*sum(max(ones(m,1)-(K*v).*y,0))+norm(v,1);
67      OFV_x=[OFV_x,valx];
68      OFV_v=[OFV_v,valv];
69      TimePerIt=[TimePerIt,timePerIteration];
70       %---------------------------------
71      totalTime=totalTime+timePerIteration;
72      tic
73  end
74  end
```

In the lines 61-67 of the code *StochasticAlgV1* and in the lines 63-69 of the program *StochasticAlgV2* the output arguments are computed each after a certain amount of iterations and added in a list in order to generate the figures in the preceding section. The time needed therefore is excluded of the time the programs needed to perform a iteration in order do avoid falsifying of the measurements. Usually it is enough to compute the output arguments just once after the last iteration.

The following sub-program basically performs the process described in Section 4.2.2, i.e. splitting the Gram matrix according to the batches defined by Eq. 116.

```
function [ Block, epsilon ] = MatrixBlocking( M, batchSize )
% M is a m by m matrix, batchSize a vector with Dimension <= m
% Block and epsilon are lists of matrices and vectors, resp.
    m=size(M, 1);
    Step=ceil(m/batchSize);
    Block=cell(1,Step);
    epsilon=cell(1,Step);
    range=randperm(m);
    for i=1:Step
        l=min(batchSize*i,m);
        epsilon{i}=sort(range(batchSize*(i-1)+1:l))';
        Block{i}=M(epsilon{i},:);
    end
end
```

The following sub-functions compute the proximal points to the conjugate function of the hinge loss and to the 1-norm according to the Eqs. 113 and 114, respectively.

```
function [ z ] = proxConjugateHingeLoss( x, y, gamma, C )
% Proximal Point Operator of the Conjungate Function
% of the Hinge Loss gamma*C*max(1-x*y,0).
% Dimension of x = Dim. of y = Dim. of z = a positive integer.
% Dimension of gamma = Dimension of C = 1.
    z=max(min(x.*y-gamma,0),-C).*y;
end
```

```
function [ z ] = proxNorm1( x, gamma )
% Proximal Point Operator of gamma*||.||_1.
% Dimension of gamma = 1.
% Dimension of x = Dimension of z = a positive integer.
    z=zeros(size(x,1),1);
    greater=x>gamma;
    z(greater)=x(greater)-gamma;
    lower=x<-gamma;
    z(lower)=x(lower)+gamma;
end
```

# References


[1] Combettes, P.L., Pesquet J.-C.: Stochastic Quasi-Fejér Block-Coordinate Fixed Point Iterations with Random Sweeping. SIAM Journal on Optimization (SIOPT) Vol. 25, No. 2, pp. 1221-1248, (2015).

[2] Combettes, P.L., Pesquet J.-C.: Primal-Dual Splitting Algorithm for Solving Inclusions with Mixtures of Composite, Lipschitzian, and Parallel-Sum Type Monotone Operators. Set-Valued Var. Anal (2012) 20:307-330.

[3] Bauschke, H.H., Combettes, P.L.: Convex Analysis and Monotone Operator Theory in Hilbert Spaces. Springer, New York (2011).

[4] Loève, M.: Probability Theory II, 4th ed. Springer, New York (1978).

[5] Briceño-Arias, L.M., Combettes, P.L.: A Monotone + Skew Splitting Model for Composite Monotone Inclusions in Duality. SIAM Journal on Optimization (SIOPT), (2011).

[6] Fortet, R.M.: Vecteurs, Fonctions et Distributions Alèatoires dans les Espaces de Hilbert. Hermès, Paris, (1995).

[7] Ledoux, M., Talagrand, M.: Probability in Banach Spaces: Isoperimetry and Processes. Springer, New York, (1991).

[8] Boţ R.I., Csetnek E.R., Hendrich C.: Recent Developments on Primal-Dual Splitting Methods with Applications to Convex Minimization. In: Pardalos P., Rassias T. (eds) Mathematics Without Boundaries. Springer, New York, (2014).

[9] Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge University Press. Cambridge, (2004).

[10] Beck, A.: First-Order Methods in Optimization. SIAM, (2017).

[11] Zălinescu, C.: Convex Analysis in General Vector Spaces. World Scientific, River Edge, NJ, (2002).

# 5 Abstract / Zusammenfassung


### Abstract

We propose a stochastic primal-dual forward-backward-forward algorithm for solving monotone inclusion problems involving maximally monotone operators, linear compositions of parallel sums of maximally monotone operators and single-valued Lipschitzian monotone operators. The stochastic algorithm is expected to converge faster than its deterministic counterpart, since every iteration computational effort is saved by using a random sweeping strategy. Based on the concept of quasi-Fejér monotonicity, we prove the almost sure convergence of the algorithm. In the second half of this thesis, we discuss the employment of the proposed algorithm in context of solving convex minimization problems, arising in the field of Kernel based Machine Learning. We use a pool of hand-written images showing the numbers four or five in order to train a Support Vector Machine, which aims to classify the images correctly. This issue is equivalent to solving a convex minimization problem. By means of this application we test several types of the stochastic algorithm, assess their performances and compare them to the performance of its deterministic counterpart. Finally, we present the according MATLAB-codes.



### Zusammenfassung

Wir präsentieren ein stochastisches Primal-duales Vorwärts-Rückwärts-Vorwärts Verfahren zum Lösen des monotonen Inklusions-Problems, welches maximal- monotone Operatoren, lineare Zusammensetzungen paralleler Summen von maximal- monotonen Operatoren und einzelwertige Lipschitz-stetige monotone Operatoren umfasst. Wir erwarten uns ein besseres Konvergenzverhalten vom stochastischen Algorithmus im Vergleich zu seinem deterministischen Gegenüber, weil ersterer durch zufälliges Aktivieren der Koordinaten in jeder Iteration Rechenaufwand einspart. Basierend auf dem Konzept der quasi-Fejér Monotonie beweisen wir die fast-sicher Konvergenz des Algorithmuses. Im zweiten Teil dieser Arbeit zeigen wir, wie man das stochastische Primal-duale Vorwärts-Rückwärts-Vorwärts Verfahren einerseits zum Lösen allgemeiner konvexer Optimierungsaufgaben und darauf basierend andererseits zum Lösen speziell für das Kernel-basierte Maschinelle Lernen verwenden kann. Konkret benützen wir eine große Auswahl von Bildern, welche die handgeschriebenen Ziffern Vier oder Fünf zeigen, um eine Support Vektor Maschine zu trainieren. Diese zielt darauf ab, die Bilder richtig zu unterscheiden. Abstrakt formuliert gleicht diese Aufgabe einer konvexen Optimierungsaufgabe. Anhand dieser Anwendung testen wir verschiedene Typen des stochastischen Algorithmuses, bewerten deren Leistungsfähig- keiten und vergleichen diese mit der Leistungsfähigkeit seines deterministischen Gegenüber. Zum Schluss präsentieren wir noch die MATLAB-Codes.