



# Faster randomized block sparse Kaczmarz by averaging

Lionel Tondji<sup>1</sup> · Dirk A. Lorenz<sup>1</sup>

Received: 6 April 2022 / Accepted: 29 November 2022 / Published online: 28 December 2022  
© The Author(s) 2022

## Abstract

The standard randomized sparse Kaczmarz (RSK) method is an algorithm to compute sparse solutions of linear systems of equations and uses sequential updates, and thus, does not take advantage of parallel computations. In this work, we introduce a parallel (mini batch) version of RSK based on averaging several Kaczmarz steps. Naturally, this method allows for parallelization and we show that it can also leverage large overrelaxation. We prove linear expected convergence and show that, given that parallel computations can be exploited, the method provably provides faster convergence than the standard method. This method can also be viewed as a variant of the linearized Bregman algorithm, a randomized dual block coordinate descent update, a stochastic mirror descent update, or a relaxed version of RSK and we recover the standard RSK method when the batch size is equal to one. We also provide estimates for inconsistent systems and show that the iterates converges to an error in the order of the noise level. Finally, numerical examples illustrate the benefits of the new algorithm.

**Keywords** Randomized Kaczmarz · Sparse solutions · Parallel methods

**Mathematics Subject Classification (2010)** 65F10 · 68W20 · 68W10 · 90C25

## 1 Introduction

In this work, we are concerned with the fundamental problem of approximating sparse solutions of large-scale linear systems of the form

$$Ax = b \tag{1.1}$$

---

✉ Dirk A. Lorenz  
d.lorenz@tu-braunschweig.de

with matrix  $A \in \mathbb{R}^{m \times n}$  and right hand side  $b \in \mathbb{R}^m$ . Linear systems like (1.1) arise in several fields of engineering and physics problems, such as sensor networks [47], signal processing [14], partial differential equations [36], filtering [20], computerized tomography [17], optimal control [37], inverse problems [18, 40], and machine learning. When  $A$  is too large to fit in memory, direct methods for solving (1.1) are not feasible and iterative methods are preferred. As long as one can afford full matrix vector products or the system matrix fits in memory, Krylov methods including the conjugate gradient (CG) algorithms [45] are the industrial standard. On the other hand, randomized methods such as the randomized (block) Kaczmarz [19, 46] and coordinate descent method [35] are effective if a single matrix vector product is too expensive and in some situations are even more efficient than CG method (see, e.g., [46] for an example). Linear convergence of the Kaczmarz method has been shown in the randomized case in [46] and [39] analyzes convergence rates in the deterministic case. Moreover, block Kaczmarz methods [28, 29, 31, 38, 42] have received much attention for their high efficiency for solving (1.1) and distributed implementations. In this work, we propose and analyze the randomized sparse Kaczmarz method [43] and show that parallel computations and averaging as in [28] lead to faster convergence.

### 1.1 Related work

**Randomized Kaczmarz** In the large data regime, the randomized Kaczmarz (RK) is a popular iterative method for solving linear systems. In each iteration,<sup>1</sup> a row vector  $a_i^T$  of  $A$  is chosen at random from the system (1.1) and the current iterate  $x_k$  is projected onto the solution space of that equation to obtain  $x_{k+1}$ . Geometrically, at each iteration

$$x_{k+1} = \operatorname{argmin}_{x \in \mathbb{R}^n} \|x - x_k\|_2^2 \quad \text{s.t.} \quad \langle a_i, x \rangle = b_i.$$

It has been observed that the convergence of RK method can be accelerated by introducing relaxation. In a relaxed variant of RK, a step is taken in the direction of this projection with the size of the step depending on a relaxation parameter. Explicitly, the relaxed RK update is given by

$$x_{k+1} = x_k - w_{k,i} \frac{\langle a_i, x_k \rangle - b_i}{\|a_i\|_2^2} \cdot a_i, \tag{1.2}$$

with initial values  $x_0 = 0$ , where the  $w_{k,i}$ ,  $i \in \{1, \dots, m\}$  are relaxation parameters. Note that this update rule requires low cost per iteration and storage of order  $\mathcal{O}(n)$ . For consistent systems, the relaxation parameters must satisfy

$$0 < \liminf_{k \rightarrow \infty} w_{k,i} \leq \limsup_{k \rightarrow \infty} w_{k,i} < 2 \tag{1.3}$$

to ensure convergence [15]. Fixing the relaxation parameters  $w_{k,i} = 1$  for all iterations  $k$  and indices  $i$  leads to the standard RK method. In [29], a block Kaczmarz

---

<sup>1</sup>We use subscript indices for components of a vector, columns or rows of a matrix, and also as iteration indices. But the meaning should always be clear from the context.

variant under the name randomized block Kaczmarz (RBK) has been analyzed. Linear convergence in expectation was shown for consistent systems of equations, with a rate depending on the geometric properties of the matrix, its submatrices, and on the size of the blocks. The convergence rate given in [29] depends on the block size and the stochastic conditioning parameter of the most ill-conditioned block of the partition when a partition is used and on the most ill-conditioned block of the entire matrix  $\mathbf{A}$  when the indices are sampled i.i.d. with replacement. The paper [31] considers more general sampling strategies such as sampling from a partition of the rows of the matrix. In [10], the authors investigate an extension of the randomized averaged block Kaczmarz method that can solve least squares problems. A parallel version of RK where a weighted average of independent updates is used was studied in [28]. They showed that as the number of threads increases, the rate of convergence improves and the convergence horizon for inconsistent systems decreases. Another more general class of block methods are sketch-and-project methods [13, 42]. For a linear system  $\mathbf{Ax} = b$ , sketch-and-project methods iteratively project the current iterate onto the solution space of a sketched subsystem  $\mathbf{S}^T \mathbf{Ax} = \mathbf{S}^T b$ . In particular, RK is a sketch-and-project method with  $\mathbf{S}$  being rows of the identity matrix.

**Randomized sparse Kaczmarz** Recently, a new variant of the standard RK method, namely the randomized sparse Kaczmarz method (RSK) [24, 38, 43] with almost the same low cost and storage requirements has shown good performance in approximating sparse solutions of large consistent linear systems. It uses two variables  $x_k^*$  and  $x_k$  and the relaxed RSK update is given by

$$\begin{aligned}
 x_{k+1}^* &= x_k^* - w_{k,i} \frac{\langle a_i x_k \rangle - b_i}{\|a_i\|_2^2} \cdot a_i, \\
 x_{k+1} &= \mathcal{S}_\lambda(x_{k+1}^*)
 \end{aligned}
 \tag{1.4}$$

with initial values  $x_0 = x_0^* = 0$ ,  $\lambda > 0$ , and the soft shrinkage operator

$$\mathcal{S}_\lambda(x) = \max\{|x| - \lambda, 0\} \cdot \text{sign}(x).$$

Fixing the relaxation parameters  $w_{k,i} = 1$  for all iterations  $k$  and indices  $i$  lead to the standard RSK method. For consistent systems, the iterates of the standard RSK method converge in expectation to the solution of the regularized *Basis Pursuit Problem*

$$\min_{x \in \mathbb{R}^n} \lambda \cdot \|x\|_1 + \frac{1}{2} \cdot \|x\|_2^2 \quad \text{s.t.} \quad \mathbf{Ax} = b.
 \tag{1.5}$$

The advantage of (1.5) is the strong convexity property of the objective function. It is important to note that (see [11]) for a large but finite parameter  $\lambda > 0$ , the solution of (1.5) gives a solution of

$$\min_{x \in \mathbb{R}^n} \|x\|_1 \quad \text{s.t.} \quad \mathbf{Ax} = b,
 \tag{1.6}$$

which is the famous *Basis Pursuit Problem* [6]. The  $\ell_1$ -norm has been used in many applications, with the goal of obtaining sparse or even sparsest solutions of

underdetermined systems of linear equations and least-squares problems which is the basis of the theory of compressed sensing [4, 7]. The sparsest solution is given by minimizing the so-called zero-norm,  $\|x\|_0$  counting the number of nonzero components in  $x$ . However, it is computationally intractable even for the simplest instances due to its combinatorial nature (it is even strongly NP-complete, see problem [MP5] in the seminal reference [12]). In [5, 8], reasonable conditions are given under which a solution of (1.6) is a sparsest solution. In [44], an extension of the RSK with linear expected convergence has been proposed for solving sparse least squares and impulsive noise problems while requiring only one additional column of the system matrix in each iteration.

**Block sparse Kaczmarz methods** In this setting, a subset of rows  $\mathbf{A}_{\tau_k}$  is used at each iteration, with  $\tau_k \subseteq \{1, \dots, m\}$  and  $|\tau_k| > 1$  where  $|\tau_k|$  denotes the cardinality of the set of indices  $\tau_k$ . We usually have two approaches. The first variant is simply a *block generalization* of the basic sparse Kaczmarz and the update is given by

$$\begin{aligned}
 x_{k+1}^* &= x_k^* - w_{k,i} \frac{\mathbf{A}_{\tau_k}^T (\mathbf{A}_{\tau_k} x_k - b_{\tau_k})}{\|\mathbf{A}_{\tau_k}\|_2^2}, \\
 x_{k+1} &= S_\lambda(x_{k+1}^*)
 \end{aligned}
 \tag{1.7}$$

Such block variants are considered, e.g., in [29, 31] (with  $\lambda = 0$ ) and in [23, 38] for  $\lambda \geq 0$  and we refer to these iterative process as *block sparse Kaczmarz* method. When  $|\tau_k| = m$ , we refer to (1.7) as the *linearized Bregman method* [2, 23, 48] and when  $|\tau_k| = 1$  as the *randomized sparse Kaczmarz method* [24, 43]. The main drawback of (1.7) is that it is not adequate for distributed implementations. The second variant of block sparse Kaczmarz can take advantage of distributed computing: each iteration takes  $\eta$  steps of the relaxed randomized sparse Kaczmarz, independently in parallel, averages the results and applies the soft shrinkage to form the next iterate. This leads to the following iteration:

$$\begin{aligned}
 x_{k+1}^* &= x_k^* - \frac{1}{\eta} \sum_{i \in \tau_k} w_i \frac{\langle a_i, x_k \rangle - b_i}{\|a_i\|_2^2} \cdot a_i, \\
 x_{k+1} &= S_\lambda(x_{k+1}^*)
 \end{aligned}
 \tag{1.8}$$

with initial values  $x_0 = x_0^* = 0$ , where  $\tau_k \subseteq \{1, \dots, m\}$  denotes a random set of  $\eta$  row indices sampled with replacement (and the  $i$ th row is chosen with probability  $p_i$ ) and  $w_i$  represents the weight corresponding to the  $i$ th row. Such block variants are considered, e.g., in [13, 27–29, 42] (with  $\lambda = 0$ ). Method (1.8) is the main method presented and analyzed in this paper and we refer to it as the *randomized sparse Kaczmarz with averaging* (RSKA) with more details in Algorithm 1. Note that the update (1.8) is easy to implement on distributed computing units, and it is comparable in terms of cost per iteration to the basic sparse Kaczmarz update, i.e., of order  $\mathcal{O}(\eta n)$ . If  $\tau_k$  is a set of one index, we recover the relaxed RSK method and if in addition the weights are chosen as  $w_i = 1$  for  $i \in \{1, \dots, m\}$ , we recover the standard RSK method.

**Require:** starting points  $x_0 = x_0^* = 0 \in \mathbb{R}^n$ , matrix  $\mathbf{A} \in \mathbb{R}^{m \times n}$  with rows  $0 \neq a_i^T \in \mathbb{R}^n$  and vector  $b \in \mathbb{R}^m$ , batch size  $\eta$ , weights  $\{w_i\}_{i=1}^m \in \mathbb{R}$  and probabilities  $p_i$

**Ensure:** (approximate) solution of  $\min_{x \in \mathbb{R}^n} \lambda \|x\|_1 + \frac{1}{2} \|x\|_2^2$  s.t.  $\mathbf{A}x = b$

1: initialize  $k = 0$

2: **repeat**

3:  $\tau_k \leftarrow \eta$  indices sampled from  $\{1, \dots, m\}$

4: Compute  $\delta_k = \frac{1}{\eta} \sum_{i \in \tau_k} w_i \frac{(a_i, x_k) - b_i}{\|a_i\|_2^2} \cdot a_i$  ▷ In parallel

5: Update  $x_{k+1}^* = x_k^* - \delta_k$

6: Update  $x_{k+1} = S_\lambda(x_{k+1}^*)$

7: Increment  $k = k + 1$

8: **until** a stopping criterion is satisfied

**Algorithm 1** Randomized sparse Kaczmarz with averaging (RSKA).

### 1.2 Contribution and organization

To the best of our knowledge, the proposed block variant (1.8) has not yet been proposed and analyzed for the randomized sparse Kaczmarz method. In this work, we make the following contributions:

- We propose a mini batch version termed *RSKA* of the randomized sparse Kaczmarz method, a general algorithm that unifies a variety of other methods such as the randomized Kaczmarz and the randomized sparse Kaczmarz with their relaxed variants. It is theoretically well-motivated, can exploit parallel computation, and converges linearly in expectation.
- We prove that our proposal leads to faster convergence than its standard counterpart. We also validate this empirically and we provide implementations of our algorithm in Python.

The remainder of the paper is organized as follows. Section 2 provides a brief overview on convexity and Bregman distances. In Section 3, we give several interpretations of our method. Section 4 provides convergence guarantees for our proposed method. In Section 5, numerical experiments demonstrate the effectiveness of *RSKA* and provides several insights regarding its behavior and its hyper-parameters. Finally, Section 6 draws some conclusions.

### 1.3 Notation

In this section, we introduce notation that will be used throughout. The first  $m$  integers are denoted by  $[m] \stackrel{\text{def}}{=} \{1, 2, \dots, m\}$ . Given a symmetric positive definite matrix  $\mathbf{B}$ , we equip the space  $\mathbb{R}^n$  with the Euclidean inner product defined by

$$\langle x, y \rangle_{\mathbf{B}} \stackrel{\text{def}}{=} \langle x, \mathbf{B}y \rangle = \sum_{i,j \in [n]} x_i \mathbf{B}_{ij} y_j, \quad x, y \in \mathbb{R}^n$$

We also define the induced norm:  $\|\cdot\|_{\mathbf{B}}^2 \stackrel{\text{def}}{=} \langle \cdot, \cdot \rangle_{\mathbf{B}}$  and use the short-hand notation  $\|\cdot\|$  to mean  $\|\cdot\|_{\mathbf{I}}$  to denote the standard 2-norm.

Let  $\mathbf{A} \in \mathbb{R}^{m \times n}$  be a real matrix. By **Range**( $\mathbf{A}$ ),  $\|\mathbf{A}\|_F$ , and  $a_i^T$ , we denote its range space, Frobenius norm, and  $i$ th row respectively, and by  $A^\dagger$ , we denote the (Moore-Penrose) pseudo-inverse. The indicator function of a set  $C$  is denoted by

$$\delta_C(x) \stackrel{\text{def}}{=} \begin{cases} 0, & \text{if } x \in C \\ +\infty, & \text{if } x \notin C \end{cases}$$

By  $e_i$ , we denote the  $i$ th column of the identity matrix  $\mathbf{I}_n \in \mathbb{R}^{n \times n}$ . For a random vector  $x_i$  that depends on a random index  $i \in [q]$  (where  $i$  is chosen with probability  $p_i$ ), we denote  $\mathbb{E}_{i \sim p}[x_i] \stackrel{\text{def}}{=} \sum_{i \in [q]} p_i x_i$  and we will just write  $\mathbb{E}[x_i]$  when the probability distribution is clear from the context. Let  $\sigma_i(\mathbf{A})$  be the  $i$ th singular value of  $A$  (when ordered decreasingly) and  $\sigma_{\min}(\mathbf{A})$  and  $\sigma_{\max}(\mathbf{A})$  be the smallest and largest singular values of  $\mathbf{A}$ , respectively. They are given by

$$\sigma_{\min}(\mathbf{A}) \stackrel{\text{def}}{=} \min_{x \in \mathbb{R}^n, x \neq 0} \frac{\|\mathbf{A}x\|_2}{\|x\|_2} \quad \text{and} \quad \sigma_{\max}(\mathbf{A}) \stackrel{\text{def}}{=} \sigma_1(\mathbf{A}) \stackrel{\text{def}}{=} \max_{x \in \mathbb{R}^n, x \neq 0} \frac{\|\mathbf{A}x\|_2}{\|x\|_2} \tag{1.9}$$

Finally, a result we will need, if  $\mathbf{A}$  is a symmetric positive semi-definite matrix, the largest singular value of  $\mathbf{A}$  (the  $L_2$  induced matrix norm or the spectral norm) can be defined instead as

$$\|\mathbf{A}\|_2 \stackrel{\text{def}}{=} \sigma_{\max}(\mathbf{A}) = \max_{x \in \mathbb{R}^n, x \neq 0} \frac{|\langle \mathbf{A}x, x \rangle|}{\|x\|_2^2} = \max_{x \in \mathbb{R}^n, x \neq 0} \frac{\|\mathbf{A}x\|_2}{\|x\|_2}. \tag{1.10}$$

Thus clearly

$$\frac{|\langle x, x \rangle_{\mathbf{A}}|}{\|x\|_2^2} \leq \sigma_{\max}(\mathbf{A}).$$

Let  $\tilde{\sigma}_{\min}(\mathbf{A}) \stackrel{\text{def}}{=} \min\{\sigma_{\min}(\mathbf{A}_J) \mid J \subseteq [n], \mathbf{A}_J \neq 0\}$  where  $\mathbf{A}_J$  denotes the submatrix of  $\mathbf{A}$  that is built up by the columns indexed by  $J$  and  $|x|_{\min} \stackrel{\text{def}}{=} \min\{|x_j| \mid x_j \neq 0\}$ .

## 2 Basic notions

At first we recall some well-known concepts and properties of convex functions and Bregman distances and later give upper-bounds of singular values of sum of matrices. Let  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  be convex (note that we assume that  $f$  is finite everywhere, hence also continuous). The *subdifferential* of  $f$  is defined by

$$\partial f(x) \stackrel{\text{def}}{=} \{x^* \in \mathbb{R}^n \mid f(y) \geq f(x) + \langle x^*, y - x \rangle \text{ for all } y \in \mathbb{R}^n\}$$

at any  $x \in \mathbb{R}^n$  is nonempty, compact and convex.

The function  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  is said to be  $\alpha$ -strongly convex, if for all  $x, y \in \mathbb{R}^n$  and subgradients  $x^* \in \partial f(x)$ , we have

$$f(y) \geq f(x) + \langle x^*, y - x \rangle + \frac{\alpha}{2} \cdot \|y - x\|_2^2.$$

If  $f$  is  $\alpha$ -strongly convex, then  $f$  is coercive, i.e.,

$$\lim_{\|x\|_2 \rightarrow \infty} f(x) = \infty,$$

and its *Fenchel conjugate*  $f^* : \mathbb{R}^n \rightarrow \mathbb{R}$  given by

$$f^*(x^*) \stackrel{\text{def}}{=} \sup_{y \in \mathbb{R}^n} \langle x^*, y \rangle - f(y)$$

is also convex, finite everywhere, and coercive.

Additionally,  $f^*$  is differentiable with a *Lipschitz-continuous gradient* with constant  $L_{f^*} = \frac{1}{\alpha}$ , i.e., for all  $x^*, y^* \in \mathbb{R}^n$ , we have

$$\|\nabla f^*(x^*) - \nabla f^*(y^*)\|_2 \leq L_{f^*} \cdot \|x^* - y^*\|_2,$$

which implies the estimate

$$f^*(y^*) \leq f^*(x^*) - \langle \nabla f^*(x^*), y^* - x^* \rangle + \frac{L_{f^*}}{2} \cdot \|x^* - y^*\|_2^2. \tag{2.1}$$

*Example 2.1* The objective function

$$f(x) \stackrel{\text{def}}{=} \lambda \cdot \|x\|_1 + \frac{1}{2} \cdot \|[2]x\|_2^2 \tag{2.2}$$

is strongly convex with constant  $\alpha = 1$  and its conjugate function can be computed with the soft shrinkage operator as

$$f^*(x^*) = \frac{1}{2} \cdot \|S_\lambda(x^*)\|_2^2 \quad \text{with} \quad \nabla f^*(x^*) = S_\lambda(x^*).$$

**Definition 2.2** The *Bregman distance*  $D_f^{x^*}(x, y)$  between  $x, y \in \mathbb{R}^n$  with respect to  $f$  and a subgradient  $x^* \in \partial f(x)$  is defined as

$$D_f^{x^*}(x, y) \stackrel{\text{def}}{=} f(y) - f(x) - \langle x^*, y - x \rangle.$$

Fenchel’s equality states that  $f(x) + f^*(x^*) = \langle x, x^* \rangle$  if  $x^* \in \partial f(x)$  and implies that the Bregman distance can be written as

$$D_f^{x^*}(x, y) = f^*(x^*) - \langle x^*, y \rangle + f(y).$$

*Example 2.3* (cf. [43]) For  $f(x) = \frac{1}{2} \cdot \|x\|_2^2$ , we just have  $\partial f(x) = \{x\}$  and  $D_f^{x^*}(x, y) = \frac{1}{2} \|x - y\|_2^2$ . For  $f(x) = \lambda \cdot \|x\|_1 + \frac{1}{2} \cdot \|x\|_2^2$  and any  $x^* = x + \lambda \cdot s \in \partial f(x)$ , we have

$$D_f^{x^*}(x, y) = \frac{1}{2} \cdot \|x - y\|_2^2 + \lambda \cdot (\|y\|_1 - \langle s, y \rangle).$$

The following properties are crucial for the convergence analysis of the randomized algorithms. They immediately follow from the definition of the Bregman

distance and the assumption of strong convexity of  $f$ , cf. [23]. For all  $x, y, z \in \mathbb{R}^n$  and  $x^* \in \partial f(x), y^* \in \partial f(y), z^* \in \partial f(z)$ , we have

$$\frac{\alpha}{2} \|x - y\|_2^2 \leq D_f^{x^*}(x, y) \leq \langle x^* - y^*, x - y \rangle \leq \|x^* - y^*\|_2 \cdot \|x - y\|_2 \tag{2.3}$$

$$D_f^{x^*}(x, y) + D_f^{y^*}(y, z) - D_f^{x^*}(x, z) = \langle x^* - y^*, z - y \rangle \tag{2.4}$$

Note that if  $f$  is differentiable with a Lipschitz-continuous gradient, then we also have the (better) upper estimate  $D_f^{x^*}(x, y) \leq L_f \cdot \|x - y\|_2^2$ , but in general, this needs not be the case.

The following Theorem will be use in the convergence analysis more precisely in Lemma (4.7).

**Theorem 2.4** ([16, Theorem 3.3.16(c)]) *Let  $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m \times n}$  be given and let  $p = \min\{m, n\}$ . Then it holds for the decreasingly ordered singular values of  $\mathbf{A}, \mathbf{B}, \mathbf{A} + \mathbf{B}$  that*

$$|\sigma_i(\mathbf{A} + \mathbf{B}) - \sigma_i(\mathbf{A})| \leq \sigma_1(\mathbf{B}), \quad \text{for } i \in [p].$$

*In particular, we have*

$$\sigma_i(\mathbf{A} + \mathbf{B}) \geq \sigma_i(\mathbf{A}) - \sigma_1(\mathbf{B}), \quad \text{for } i \in [p].$$

### 3 Interpretations

We can view the randomized sparse Kaczmarz with averaging algorithm as an optimization method for solving a specific primal or dual optimization problem. More precisely, the RSKA algorithm is a particular case of the following.

#### 3.1 Randomized block/parallel coordinate descent

Considering the regularized Basis Pursuit Problem as primal problem

$$\min_{x \in \mathbb{R}^n} \lambda \cdot \|x\|_1 + \frac{1}{2} \cdot \|x\|_2^2 \quad \text{s.t. } \mathbf{A}x = b. \tag{3.1}$$

The dual of optimization problem (3.1) takes the form of a quadratic program:

$$\min_{y \in \mathbb{R}^m} \frac{1}{2} \cdot \|S_\lambda(\mathbf{A}^T y)\|_2^2 - \langle b, y \rangle. \tag{3.2}$$

where the primal variable  $x$  and the dual variable  $y$  are related through the relation  $x = S_\lambda(\mathbf{A}^T y)$ . Let us define the primal and dual objective functions

$$f(x) = \lambda \cdot \|x\|_1 + \frac{1}{2} \cdot \|x\|_2^2 + \delta_{\{0\}}(b - \mathbf{A}x)$$

and

$$g(y) = \frac{1}{2} \cdot \|S_\lambda(\mathbf{A}^T y)\|_2^2 - \langle b, y \rangle,$$

respectively. One iteration of the RSKA algorithm can be viewed as one step of the randomized block coordinate descent (RBCD) applied to the dual problem (3.2) when the weights  $w_i$  are chosen in a particular form [38]. More formally a negative gradient



step in the random  $i$ th component of  $y$  having  $\nabla_{i_k} g(y) = a_{i_k}^T S_\lambda(\mathbf{A}^T y) - b_{i_k}$  with step size  $t_k = \frac{1}{\|a_{i_k}\|_2^2}$  yields

$$y_{k+1} = y_k - \frac{1}{\|a_{i_k}\|_2^2} \nabla_{i_k} g(y_k) e_{i_k}$$

We easily recover (1.4) by simply multiplying this update with  $\mathbf{A}^T$  and using the relation between the primal and dual variables given by  $x = S_\lambda(\mathbf{A}^T y)$ . Consider the dual problem (3.2). If we choose the particular weights  $w_{k,i} = \frac{\|a_i\|_2^2}{\sum_{i \in \tau_k} \|a_i\|_2^2}$ , the block coordinate descent method applied to  $g$  from (3.2) reads

$$x_{k+1}^* = x_k^* - \frac{1}{\eta \sum_{i \in \tau_k} \|a_i\|_2^2} \sum_{i \in \tau_k} ((a_i, x_k) - b_i) \cdot a_i,$$

$$x_{k+1} = S_\lambda(x_{k+1}^*)$$

Parallel coordinate descent [41] applied to the dual problem (3.2) with learning rate  $t_k = \frac{1}{\eta \|a_i\|_2^2}$  yields

$$y_{k+1} = y_k - \sum_{i \in \tau_k} \frac{1}{\eta \|a_i\|_2^2} \nabla_i g(y_k) e_i \tag{3.3}$$

In Update (3.3), only coordinate  $i \in \tau_k$  are updated in  $y_{k+1}$  and the remaining coordinates are unchanged. Multiplying this update with  $\mathbf{A}^T$  and using the relation between the primal and dual variables, we recover (1.4) with particular weights  $w_i = 1$ . However, for general weights  $w_k$ , the RSKA algorithm cannot be interpreted in these ways, and thus our scheme is more general. It is important to note that in [38] for the randomized block sparse Kaczmarz method of type (1.7), sublinear convergence rates have been obtained by identifying the iteration as a randomized block coordinate gradient descent method applied to the objective function  $g$  of the unconstrained dual of  $f$ . However, the rates given in [38] are in terms of the dual objective function  $g$ , and not of the primal iterates only, although, as mentioned there in the conclusions, the experimental results indicate that such rates also hold for the primal iterates.

### 3.2 Stochastic mirror descent with stochastic Polyak stepsize

The stochastic mirror descent (SMD) method and its variants [1, 21, 32] is one of the most widely used family of algorithms in stochastic optimization for non-smooth, Lipschitz continuous—convex and non-convex functions. Starting with the original work of [34], SMD has been studied in the context of convex programming [33], saddle-point problems [25], and monotone variational inequalities [26]. Now we draw a connection of Algorithm 1 to the stochastic mirror descent method using stochastic Polyak stepsizes. We consider a set of sketching matrices  $\mathbf{S}_i \in \mathbb{R}^{m \times s}$  ( $i \in [m]$ ), define  $\mathbf{Z}_i \stackrel{\text{def}}{=} \mathbf{S}_i(\mathbf{S}_i^T \mathbf{A} \mathbf{A}^T \mathbf{S}_i)^\dagger \mathbf{S}_i^T$  and consider the stochastic convex quadratic  $h_{\mathbf{S}_i}$

$$h_{\mathbf{S}_i}(x) \stackrel{\text{def}}{=} \frac{1}{2} \|\mathbf{A}x - b\|_{\mathbf{Z}_i}^2 = \frac{1}{2} (\mathbf{A}x - b)^T \mathbf{Z}_i (\mathbf{A}x - b), \tag{3.4}$$

(recall that  $\mathbf{A} \in \mathbb{R}^{m \times n}$ ,  $b \in \mathbb{R}^m$ ).

The general sketched mirror descent update with learning rate  $t_k$  and a mirror map  $f$  works as follows: For a given iterate  $x_k$  draw a sketching matrix  $\mathbf{S}_{i_k}$  (actually, one draws an index  $i_k$ ) at random and update

$$x_{k+1} = \arg \min_{x \in \mathbb{R}^n} \left\{ \langle \nabla h_{\mathbf{S}_{i_k}}(x_k), x - x_k \rangle + \frac{1}{t_k} D_f^{x_k^*}(x_k, x) \right\}, \quad x_k^* \in \partial f(x_k), \quad (3.5)$$

which yields to the following update:

$$\begin{aligned} x_{k+1}^* &= x_k^* - t_k \nabla h_{\mathbf{S}_{i_k}}(x_k), \\ x_{k+1} &= \nabla f^*(x_{k+1}^*). \end{aligned} \quad (3.6)$$

where  $f^*$  denotes the Fenchel conjugate of  $f$ .

One can show that

$$D_f^{x_{k+1}^*}(x_{k+1}, \hat{x}) \leq D_f^{x_k^*}(x_k, \hat{x}) - t_k \langle \nabla h_{\mathbf{S}_{i_k}}(x_k), x_k - \hat{x} \rangle + \frac{t_k^2}{2} \|\nabla h_{\mathbf{S}_{i_k}}(x_k)\|^2 \quad (3.7)$$

(actually, this also follows from Lemma 4.3 below with  $\Phi(x) = \langle \nabla h_{\mathbf{S}_{i_k}}(x_k), x - x_k \rangle$ , and a strongly convex function  $f$ ). If we select  $t_k$  such that the RHS of inequality (3.7) is minimized, we obtain

$$t_k = \frac{\langle \nabla h_{\mathbf{S}_{i_k}}(x_k), x_k - \hat{x} \rangle}{\|\nabla h_{\mathbf{S}_{i_k}}(x_k)\|^2}. \quad (3.8)$$

Since  $\nabla h_{\mathbf{S}}(x) = \mathbf{A}^T \mathbf{Z}(\mathbf{A}x - b)$ , we get (cf. [42])

$$h_{\mathbf{S}_{i_k}}(x) - h_{\mathbf{S}_{i_k}}(\hat{x}) \stackrel{h_{\mathbf{S}_{i_k}}(\hat{x})=0}{=} h_{\mathbf{S}_{i_k}}(x) = \frac{1}{2} \|\nabla h_{\mathbf{S}_{i_k}}(x)\|^2 = \frac{1}{2} \langle \nabla h_{\mathbf{S}_{i_k}}(x_k), x - \hat{x} \rangle \quad (3.9)$$

we get that the optimal step size is in fact simply

$$t_k = \frac{2[h_{\mathbf{S}_{i_k}}(x_k) - h_{\mathbf{S}_{i_k}}(\hat{x})]}{\|\nabla h_{\mathbf{S}_{i_k}}(x_k)\|^2}. \quad (3.10)$$

This quotient is known as *stochastic mirror Polyak stepsize* [9] (note that in this particular case, we always get  $t_k = 1$ ). The randomized Kaczmarz (1.4) is equivalent to one step of the stochastic mirror descent (3.5) with the mirror Polyak stepsize  $t_k$  given in (3.8), whereas the mirror Polyak stepsize in its general form [9, 22] is given by  $t_k = \frac{h_{\mathbf{S}_{i_k}}(x_k) - h_{\mathbf{S}_{i_k}}(\hat{x})}{c \|\nabla h_{\mathbf{S}}(x_k)\|^2}$ . The parameter  $0 < c \in \mathbb{R}$  in the step size is an important quantity which can be set theoretically based on the properties of the function under study. In [22], it is suggested that, for optimal convergence, one should select  $c = 1/2$  for strongly convex functions and  $c = 0.2$  for non-convex functions. Moreover, the general RSKA method (see Algorithm 1) falls into the general sketched-and-project framework in the context of mirror descent with  $\mathbf{Z} \stackrel{\text{def}}{=} \frac{1}{\eta} \sum_{j \in \tau} w_j \mathbf{S}_j (\mathbf{S}_j^T \mathbf{A} \mathbf{A}^T \mathbf{S}_j)^{\dagger} \mathbf{S}_j^T$  with  $\mathbf{S}_j = e_j$ ,  $t_k = 1$ , and  $f(x) = \lambda \cdot \|x\|_1 + \frac{1}{2} \cdot \|x\|_2^2$ . In fact, the sketch-and-project form of updates (1.8) (resp. 3.6) is given by:

$$\begin{aligned} x_{k+1}^* &= x_k^* - \mathbf{A}^T \mathbf{Z}(\mathbf{A}x_k - b), \\ x_{k+1} &= \nabla f^*(x_{k+1}^*), \end{aligned} \quad (3.11)$$

with some random  $\tau_i \subset [m]$  with cardinality  $\eta$ .

### 4 Convergence analysis

In this section, we show expected linear convergence for the randomized sparse Kaczmarz with averaging method. Before that, we give the update satisfy by the iterate  $x_k^*$  in Algorithm 1. From line 5 of Algorithm 1, it holds that:

$$\begin{aligned} x_{k+1}^* &= x_k^* - \frac{1}{\eta} \sum_{i \in \tau_k} w_i \frac{\langle a_i, x_k \rangle - b_i}{\|a_i\|_2^2} \cdot a_i \\ &= x_k^* - \frac{1}{\eta} \sum_{i \in \tau_k} w_i (e_i^T \mathbf{A})^T \cdot \frac{e_i^T (\mathbf{A}x_k - b)}{\|a_i\|_2^2} \\ &= x_k^* - \mathbf{A}^T \frac{1}{\eta} \sum_{i \in \tau_k} w_i \cdot \frac{e_i, e_i^T}{\|a_i\|_2^2} (\mathbf{A}x_k - b) \end{aligned}$$

To simplify notation, we define the following matrices.

**Definition 4.1** Let  $\mathbf{Diag}(d_1, d_2, \dots, d_m)$  denote the diagonal matrix with  $d_1, d_2, \dots, d_m$  on the diagonal. We define the following matrices:

- Weighted sampling matrix:

$$\mathbf{M}_k = \frac{1}{\eta} \sum_{i \in \tau_k} w_i \frac{e_i e_i^T}{\|a_i\|_2^2}$$

- Normalization matrix:

$$\mathbf{D} = \mathbf{Diag}(\|a_1\|, \|a_2\|, \dots, \|a_m\|)$$

so that the matrix  $\mathbf{D}^{-1} \mathbf{A}$  has rows with unit norm.

- Probability matrix:

$$\mathbf{P} = \mathbf{Diag}(p_1, p_2, \dots, p_m)$$

where  $p_j = \mathbb{P}(i = j)$ .

- Weight matrix:

$$\mathbf{W} = \mathbf{Diag}(w_1, w_2, \dots, w_m)$$

where  $w_i$  represents the weight corresponding to the  $i$  th row.

The following lemma and proof are taken from [28, Lemma 1] and we include the full proof for completeness. The lemma gives the first and second moment of the random matrices  $\mathbf{M}_k, \mathbf{A}^T \mathbf{M}_k$  respectively and will be use in our convergence analysis.

**Lemma 4.2** Let  $\mathbf{M}_k, \mathbf{P}, \mathbf{W}$ , and  $\mathbf{D}$  be defined as in Definition 4.1. Then

$$\mathbb{E}_k [\mathbf{M}_k] = \mathbf{P}\mathbf{W}\mathbf{D}^{-2} \quad \text{and}$$

$$\mathbb{E}_k \left[ (\mathbf{A}^T \mathbf{M}_k)^T \cdot (\mathbf{A}^T \mathbf{M}_k) \right] = \frac{1}{\eta} \mathbf{P}\mathbf{W}^2 \mathbf{D}^{-2} + \left(1 - \frac{1}{\eta}\right) \mathbf{P}\mathbf{W}\mathbf{D}^{-2} \mathbf{A}\mathbf{A}^T \mathbf{P}\mathbf{W}\mathbf{D}^{-2}.$$

*Proof* Let  $\mathbb{E}_i[\cdot]$  denote  $\mathbb{E}_{i \sim p}[\cdot]$ . From the definition of the weighted sampling matrix  $\mathbf{M}_k$  as the weighted average of the i.i.d. sampling matrices  $\frac{e_i e_i^T}{\|a_i\|_2^2}$ , we see that

$$\mathbb{E}_k [\mathbf{M}_k] = \mathbb{E}_k \left[ \frac{1}{\eta} \sum_{i \in \tau_k} w_i \frac{e_i e_i^T}{\|a_i\|_2^2} \right] = \mathbb{E}_i \left[ w_i \frac{e_i e_i^T}{\|a_i\|_2^2} \right] = \sum_{i=1}^m p_i w_i \frac{e_i e_i^T}{\|a_i\|_2^2} = \mathbf{P}\mathbf{W}\mathbf{D}^{-2}.$$

In the same way, we have

$$\begin{aligned} & \mathbb{E}_k \left[ (\mathbf{A}^T \mathbf{M}_k)^T \cdot (\mathbf{A}^T \mathbf{M}_k) \right] \\ &= \mathbb{E}_k \left[ \left( \frac{1}{\eta} \sum_{i \in \tau_k} w_i \frac{e_i e_i^T}{\|a_i\|_2^2} \right) \mathbf{A} \cdot \mathbf{A}^T \cdot \left( \frac{1}{\eta} \sum_{j \in \tau_k} w_j \frac{e_j e_j^T}{\|a_j\|_2^2} \right) \right] \\ &= \mathbb{E}_k \left[ \left( \frac{1}{\eta} \sum_{i \in \tau_k} w_i \frac{e_i a_i^T}{\|a_i\|_2^2} \right) \cdot \left( \frac{1}{\eta} \sum_{j \in \tau_k} w_j \frac{a_j e_j^T}{\|a_j\|_2^2} \right) \right] \\ &= \frac{1}{\eta} \mathbb{E}_i \left[ \left( w_i \frac{e_i a_i^T}{\|a_i\|_2^2} \right) \cdot \left( w_i \frac{a_i e_i^T}{\|a_i\|_2^2} \right) \right] + \left(1 - \frac{1}{\eta}\right) \mathbb{E}_i \left[ w_i \frac{e_i a_i^T}{\|a_i\|_2^2} \right] \mathbb{E}_i \left[ w_i \frac{a_i e_i^T}{\|a_i\|_2^2} \right] \\ &= \frac{1}{\eta} \mathbb{E}_i \left[ w_i^2 \frac{e_i e_i^T}{\|a_i\|_2^2} \right] + \left(1 - \frac{1}{\eta}\right) \mathbb{E}_i \left[ w_i \frac{e_i e_i^T}{\|a_i\|_2^2} \right] \mathbf{A}\mathbf{A}^T \mathbb{E}_i \left[ w_i \frac{e_i e_i^T}{\|a_i\|_2^2} \right] \\ &= \frac{1}{\eta} \mathbf{P}\mathbf{W}^2 \mathbf{D}^{-2} + \left(1 - \frac{1}{\eta}\right) \mathbf{P}\mathbf{W}\mathbf{D}^{-2} \mathbf{A}\mathbf{A}^T \mathbf{P}\mathbf{W}\mathbf{D}^{-2}, \end{aligned}$$

by separating the cases where  $i = j$  from those where  $i \neq j$  and utilizing the independence of the indices sampled in  $\tau_k$ . □

We now present convergence results for the proposed method. We start our analysis by characterizing the error bound between two consecutive iterates and the error bound between the Bregman distance of the iterates, the solution, and the residual in the following lemma.

**Lemma 4.3** Let  $f, \Phi : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$  be convex, where  $\text{dom}(f) = \mathbb{R}^n$  and  $\text{dom}(\Phi) \neq \emptyset$ . Let  $\mathcal{X} \subseteq \text{dom}(\Phi)$  be nonempty and convex,  $x_k \in \mathbb{R}^n, x_k^* \in \partial f(x_k)$ . Assume that

$$x_{k+1} \in \arg \min_{x \in \mathcal{X}} \left\{ \Phi(x) + D_f^{x_k^*}(x_k, x) \right\}.$$

Then there exist subgradient  $x_{k+1}^* \in \partial f(x_{k+1})$  such that it holds

$$\Phi(y) + D_f^{x_k^*}(x_k, y) \geq \Phi(x_{k+1}) + D_f^{x_k^*}(x_k, x_{k+1}) + D_f^{x_{k+1}^*}(x_{k+1}, y)$$

for any  $y \in \mathcal{X}$ .

*Proof* Let denote by  $J(x) = \Phi(x) + D_f^{x_k^*}(x_k, x)$ . Since  $J$  and  $\mathcal{X}$  are convex and  $x_{k+1}$  minimizes  $J$  over  $\mathcal{X}$ , there exists a subgradient  $d \in \partial J(x_{k+1})$  such that

$$\langle d, y - x_{k+1} \rangle \geq 0 \quad \forall y \in \mathcal{X}.$$

Since  $f$  is finite everywhere, we have  $\text{dom}(D_f(\cdot, u)) = \mathbb{R}^n$  for all  $u \in \mathbb{R}^n$ . Since  $\text{dom}(\Phi)$  is nonempty and convex,  $\Phi$  has nonempty relative interior. So the subgradient sum rule applies and we obtain that

$$\partial J(x_{k+1}) = \partial \Phi(x_{k+1}) + (\partial f(x_{k+1}) - x_k^*).$$

Hence, there exist subgradients  $g \in \partial \Phi(x_{k+1})$ ,  $x_{k+1}^* \in \partial f(x_{k+1})$  such that

$$\langle g + (x_{k+1}^* - x_k^*), y - x_{k+1} \rangle \geq 0 \quad \forall y \in \mathcal{X}.$$

Therefore, using the property of the subgradient and (2.4), we have for all  $y \in \mathcal{X}$

$$\begin{aligned} \Phi(y) &\geq \Phi(x_{k+1}) + \langle g, y - x_{k+1} \rangle \\ &\geq \Phi(x_{k+1}) + \langle x_k^* - x_{k+1}^*, y - x_{k+1} \rangle \\ &\stackrel{(2.4)}{=} \Phi(x_{k+1}) + D_f^{x_k^*}(x_k, x_{k+1}) - D_f^{x_k^*}(x_k, y) + D_f^{x_{k+1}^*}(x_{k+1}, y). \end{aligned}$$

□

The following lemma provides an error bound for the Bregman distance.

**Lemma 4.4** ([43]) *Let  $\tilde{\sigma}_{\min}(\mathbf{A})$  and  $|\hat{x}|_{\min}$  be defined as in Section 1.3. Then for any  $x \in \mathbb{R}^n$  with  $\partial f(x) \cap \text{Range}(\mathbf{A}^T) \neq 0$  and for all  $\hat{x} = \mathbf{A}^T y \in \partial f(x) \cap \text{Range}(\mathbf{A}^T)$ , we have*

$$D_f^{x_k^*}(x_k, \hat{x}) \leq \gamma \cdot \|\mathbf{A}x_k - b\|_2^2 \tag{4.1}$$

where

$$\gamma = \frac{1}{\tilde{\sigma}_{\min}^2(\mathbf{A})} \frac{|\hat{x}|_{\min} + 2\lambda}{|\hat{x}|_{\min}} \tag{4.2}$$

To effectively use Lemma 4.4, we need the following assumption which characterize the coupling between the weight matrix and the probability matrix.

**Assumption 1** *The weight matrix  $\mathbf{W}$  and the probability matrix  $\mathbf{P}$  are linked by the following coupling*

$$\mathbf{PWD}^{-2} = \frac{\alpha}{\|\mathbf{A}\|_F^2} \mathbf{I}$$

some scalar relaxation parameter  $\alpha > 0$ .

Assumption (1) has been used in [28] for the inconsistent case (i.e.,  $b - \mathbf{A}\hat{x} \neq 0$ ) and for  $\lambda = 0$ . We include a motivation for completeness. As the batch size  $\eta$  goes to  $\infty$  (recall that we sample with replacement), we have

$$\lim_{\eta \rightarrow \infty} \mathbf{M}_k = \mathbb{E}_{i \sim p} \left[ w_i \frac{e_i e_i^T}{\|a_i\|_2^2} \right] = \mathbf{P}\mathbf{W}\mathbf{D}^{-2}$$

Therefore, the averaged RSK update of (1.8) approaches the deterministic update:

$$\begin{aligned} x_{k+1} &= (\mathbf{I} - \mathbf{A}^T \mathbf{P}\mathbf{W}\mathbf{D}^{-2} \mathbf{A})x_k + \mathbf{A}^T \mathbf{P}\mathbf{W}\mathbf{D}^{-2} b \\ x_{k+1} - \hat{x} &= (\mathbf{I} - \mathbf{A}^T \mathbf{P}\mathbf{W}\mathbf{D}^{-2} \mathbf{A})(x_k - \hat{x}) + \mathbf{A}^T \mathbf{P}\mathbf{W}\mathbf{D}^{-2} (b - \mathbf{A}\hat{x}) \end{aligned}$$

In order to have that  $(x_{k+1} - \hat{x})$  goes to zero in the limit, we should require that this limiting error update has the zero vector as a fixed point, i.e.,

$$0 = \mathbf{A}^T \mathbf{P}\mathbf{W}\mathbf{D}^{-2} (b - \mathbf{A}\hat{x})$$

This is guaranteed if  $\mathbf{P}\mathbf{W}\mathbf{D}^{-2} = \beta \mathbf{I}$ . But for  $\lambda \neq 0$ , we do not have a bound of the form  $D_f^{x_k^*}(x_k, \hat{x}) \leq \gamma \cdot \|Ax_k - b\|_{\mathbf{P}\mathbf{W}\mathbf{D}^{-2}}^2$  that is the reason why for that case we need to assume  $\mathbf{P}\mathbf{W}\mathbf{D}^{-2} = \beta \mathbf{I}$ .

Since in this case  $\mathbf{P}\mathbf{W}^2 \mathbf{D}^{-2} = \mathbf{P}\mathbf{W}\mathbf{D}^{-2} \mathbf{W}$ , Lemma 4.2 becomes:

**Lemma 4.5** *Let  $\mathbf{M}_k, \mathbf{P}, \mathbf{W}$  and  $\mathbf{D}$  be defined as in Definition 4.1 and let Assumption 1 hold. Then*

$$\mathbb{E}_k [\mathbf{M}_k] = \frac{\alpha \mathbf{I}}{\|\mathbf{A}\|_F^2}$$

and

$$\mathbb{E}_k \left[ \mathbf{M}_k^T \mathbf{A} \mathbf{A}^T \mathbf{M}_k \right] = \frac{1}{\eta} \frac{\alpha \mathbf{W}}{\|\mathbf{A}\|_F^2} + \alpha^2 \left(1 - \frac{1}{\eta}\right) \frac{\mathbf{A} \mathbf{A}^T}{\|\mathbf{A}\|_F^4}$$

**Lemma 4.6** *Under Assumption 1, for the iterates  $x_k$  of Algorithm 1, it holds that:*

$$\mathbb{E}_k \left[ D_f^{x_{k+1}^*}(x_{k+1}, \hat{x}) \right] \leq D_f^{x_k^*}(x_k, \hat{x}) - \frac{\alpha}{\|\mathbf{A}\|_F^2} (1 - \sigma_{\max}(\mathbf{T})) \|Ax_k - b\|_2^2.$$

with

$$\mathbf{T} = \frac{1}{2\eta} \mathbf{W} + \frac{\alpha}{2} \left(1 - \frac{1}{\eta}\right) \frac{\mathbf{A} \mathbf{A}^T}{\|\mathbf{A}\|_F^2}$$

*Proof* Using Lemma 4.3 with  $f(x) = \lambda\|x\|_1 + \frac{1}{2}\|x\|_2^2$  and  $\Phi(x) = \langle \mathbf{A}^T \mathbf{M}_k(\mathbf{A}x_k - b), x - x_k \rangle$ ,  $y = \hat{x}$ , it holds that:

$$\begin{aligned} D_f^{x_k^*} (x_{k+1}, \hat{x}) &\leq D_f^{x_k^*} (x_k, \hat{x}) + \Phi(\hat{x}) - \Phi(x_{k+1}) - D_f^{x_k^*} (x_k, x_{k+1}) \\ &= D_f^{x_k^*} (x_k, \hat{x}) - \langle \mathbf{A}^T \mathbf{M}_k(\mathbf{A}x_k - b), x_k - \hat{x} \rangle \\ &\quad + \langle \mathbf{A}^T \mathbf{M}_k(\mathbf{A}x_k - b), x_k - x_{k+1} \rangle - D_f^{x_k^*} (x_k, x_{k+1}) \\ &\leq D_f^{x_k^*} (x_k, \hat{x}) - \langle \mathbf{A}^T \mathbf{M}_k(\mathbf{A}x_k - b), x_k - \hat{x} \rangle \\ &\quad + \|\mathbf{A}^T \mathbf{M}_k(\mathbf{A}x_k - b)\| \cdot \|x_k - x_{k+1}\| - \frac{1}{2}\|x_k - x_{k+1}\|^2 \\ &\leq D_f^{x_k^*} (x_k, \hat{x}) - \langle \mathbf{A}^T \mathbf{M}_k(\mathbf{A}x_k - b), x_k - \hat{x} \rangle + \frac{1}{2}\|\mathbf{A}^T \mathbf{M}_k(\mathbf{A}x_k - b)\|^2 \end{aligned}$$

We have:

$$\begin{aligned} \mathbb{E}_k \left[ \left\langle \mathbf{A}^T \mathbf{M}_k(\mathbf{A}x_k - b), x_k - \hat{x} \right\rangle \right] &= \mathbb{E}_k \left[ \left\langle \mathbf{M}_k \cdot (\mathbf{A}x_k - b), \mathbf{A}x_k - b \right\rangle \right] \\ &\stackrel{\text{Lemma 4.2}}{=} \left\langle \mathbf{A}x_k - b, \mathbf{A}x_k - b \right\rangle_{\text{PWD}^{-2}} \\ &\stackrel{\text{Lemma 4.5}}{=} \frac{\alpha}{\|\mathbf{A}\|_F^2} \|\mathbf{A}x_k - b\|_2^2 \end{aligned}$$

and with  $\mathbf{T} = \frac{1}{2\eta} \mathbf{W} + \frac{\alpha}{2} (1 - \frac{1}{\eta}) \frac{\mathbf{A}\mathbf{A}^T}{\|\mathbf{A}\|_F^2}$ , we get

$$\begin{aligned} \mathbb{E}_k \left[ \left\| \mathbf{A}^T \cdot \mathbf{M}_k \cdot (\mathbf{A}x_k - b) \right\|_2^2 \right] &= \left\langle \mathbf{A}x_k - b, \mathbb{E}_k \left[ \mathbf{M}_k^T \mathbf{A}\mathbf{A}^T \mathbf{M}_k \right] \cdot (\mathbf{A}x_k - b) \right\rangle \\ &\stackrel{\text{Lemma 4.5}}{=} \left\langle \mathbf{A}x_k - b, \left( \frac{1}{\eta} \frac{\alpha \mathbf{W}}{\|\mathbf{A}\|_F^2} + \alpha^2 (1 - \frac{1}{\eta}) \frac{\mathbf{A}\mathbf{A}^T}{\|\mathbf{A}\|_F^4} \right) \cdot (\mathbf{A}x_k - b) \right\rangle \\ &= \frac{2\alpha}{\|\mathbf{A}\|_F^2} \left\langle \mathbf{A}x_k - b, \mathbf{A}x_k - b \right\rangle_{\mathbf{T}} \\ &\leq \frac{2\alpha}{\|\mathbf{A}\|_F^2} \sigma_{\max}(\mathbf{T}) \|\mathbf{A}x_k - b\|_2^2. \end{aligned}$$

Thus, combining everything together gives us

$$\begin{aligned} & \mathbb{E}_k \left[ D_f^{x_k^*} (x_{k+1}, \hat{x}) \right] \\ & \leq D_f^{x_k^*} (x_k, \hat{x}) - \mathbb{E}_k \left[ \left\langle \mathbf{A}^T \mathbf{M}_k (\mathbf{A}x_k - b), x_k - \hat{x} \right\rangle \right] + \frac{1}{2} \mathbb{E}_k \left[ \left\| \mathbf{A}^T \mathbf{M}_k (\mathbf{A}x_k - b) \right\|_2^2 \right] \\ & \leq D_f^{x_k^*} (x_k, \hat{x}) - \frac{\alpha}{\|\mathbf{A}\|_F^2} \|\mathbf{A}x_k - b\|_2^2 + \frac{1}{2} \frac{2\alpha}{\|\mathbf{A}\|_F^2} \sigma_{\max}(\mathbf{T}) \|\mathbf{A}x_k - b\|_2^2 \\ & \leq D_f^{x_k^*} (x_k, \hat{x}) - \frac{\alpha}{\|\mathbf{A}\|_F^2} (1 - \sigma_{\max}(\mathbf{T})) \|\mathbf{A}x_k - b\|_2^2. \end{aligned}$$

□

The following lemma gives an upper and a lower bound for the largest singular value of  $\mathbf{T}$  which will be use in the convergence of RSKA iterates.

**Lemma 4.7** *Let*

$$\mathbf{T} = \frac{1}{2\eta} \mathbf{W} + \frac{\alpha}{2} \left( 1 - \frac{1}{\eta} \right) \frac{\mathbf{A}\mathbf{A}^T}{\|\mathbf{A}\|_F^2}$$

*Then the largest singular value of  $\mathbf{T}$  satisfies:*

$$\begin{aligned} \frac{1}{2\eta} \left( \sigma_{\max}(\mathbf{W}) - \frac{\alpha}{\|\mathbf{A}\|_F^2} (\eta - 1) \sigma_{\max}^2(\mathbf{A}) \right) & \leq \sigma_{\max}(\mathbf{T}) \\ & \leq \frac{1}{2\eta} \left( \sigma_{\max}(\mathbf{W}) + \frac{\alpha}{\|\mathbf{A}\|_F^2} (\eta - 1) \sigma_{\max}^2(\mathbf{A}) \right) \end{aligned}$$

*In addition, If  $\mathbf{W} = \alpha\mathbf{I}$ , then  $\mathbf{T}$  is positive semi-definite and*

$$\sigma_{\max}(\mathbf{T}) = \frac{1}{2\eta} \left( \alpha + \frac{\alpha}{\|\mathbf{A}\|_F^2} (\eta - 1) \sigma_{\max}^2(\mathbf{A}) \right)$$

*Proof* The first part of the proof follows easily from Theorem 2.2. If  $\mathbf{W} = \alpha\mathbf{I}$ , we have: If  $\lambda$  is an eigenvalue of  $\mathbf{A}\mathbf{A}^T$ , then  $\frac{\alpha}{\eta} + \frac{\alpha}{2} \left( 1 - \frac{1}{\eta} \right) \frac{\lambda}{\|\mathbf{A}\|_F^2}$  is an eigenvalue of  $\mathbf{T}$ . From this, we deduce the last equality as well as that  $\mathbf{T}$  is positive semi-definite. □

### 4.1 General convergence result

In this part, we present general convergence results for the iterates from RSKA method.

**Theorem 4.8** (Noiseless case) *Consider  $\eta > 1$ ,  $\gamma$  as defined in (4.2) (Lemma (4.4)), let Assumption 1 hold and assume that*

$$0 < \alpha < 2 \frac{(\eta - \frac{1}{2} \sigma_{\max}(\mathbf{W})) \|\mathbf{A}\|_F^2}{\sigma_{\max}(\mathbf{A})^2 (\eta - 1)}. \tag{4.3}$$



Then the random iterates  $x_k$  produced by Algorithm 1 converge in expectation with a linear rate to the unique solution  $\hat{x}$  of  $\min_{\mathbf{A}x=b} \lambda \|x\|_1 + \frac{1}{2} \|x\|_2^2$ , more precisely, with

$$q = 1 - \frac{1}{\gamma} \cdot \frac{L(\alpha)}{\|\mathbf{A}\|_F^2} \in (0, 1), \tag{4.4}$$

and

$$L(\alpha) = \alpha - \frac{\alpha}{2\eta} \left( \frac{\alpha}{\|\mathbf{A}\|_F^2} (\eta - 1) \sigma_{\max}^2(\mathbf{A}) + \sigma_{\max}(\mathbf{W}) \right),$$

it holds that

$$\begin{aligned} \mathbb{E} \left[ D_f^{x_{k+1}^*}(x_{k+1}, \hat{x}) \right] &\leq q \cdot \mathbb{E} \left[ D_f^{x_k^*}(x_k, \hat{x}) \right] \\ \mathbb{E} \left[ \|x_k - \hat{x}\|_2^2 \right] &\leq 2 \cdot q^k \cdot f(\hat{x}). \end{aligned}$$

*Proof* Combining Lemma 4.6 with (4.1) gives

$$\begin{aligned} \mathbb{E}_k \left[ D_f^{x_{k+1}^*}(x_{k+1}, \hat{x}) \right] &\leq D_f^{x_k^*}(x_k, \hat{x}) - \frac{\alpha}{\|\mathbf{A}\|_F^2 \gamma} (1 - \sigma_{\max}(\mathbf{T})) D_f^{x_k^*}(x_k, \hat{x}) \\ &\leq \left( 1 - \frac{\alpha}{\|\mathbf{A}\|_F^2 \gamma} (1 - \sigma_{\max}(\mathbf{T})) \right) D_f^{x_k^*}(x_k, \hat{x}). \end{aligned}$$

Using the rule of total expectation, we get

$$\mathbb{E} \left[ D_f^{x_{k+1}^*}(x_{k+1}, \hat{x}) \right] \leq \left( 1 - \frac{1}{\gamma} \cdot \frac{\alpha \cdot (1 - \sigma_{\max}(\mathbf{T}))}{\|\mathbf{A}\|_F^2} \right) \mathbb{E} \left[ D_f^{x_k^*}(x_k, \hat{x}) \right].$$

To get a rate  $q \in (0, 1)$ , we need that  $(1 - \sigma_{\max}(\mathbf{T})) > 0$ , i.e.,  $\sigma_{\max}(\mathbf{T}) < 1$  which holds true from (4.3). From Lemma 4.7, it follows that  $L(\alpha) \leq \alpha(1 - \sigma_{\max}(\mathbf{T}))$  and thus we get  $\mathbb{E} \left[ D_f^{x_{k+1}^*}(x_{k+1}, \hat{x}) \right] \leq q \cdot \mathbb{E} \left[ D_f^{x_k^*}(x_k, \hat{x}) \right]$ , with  $q = 1 - \frac{1}{\gamma} \cdot \frac{L(\alpha)}{\|\mathbf{A}\|_F^2}$ . The inequality in terms of  $\|x_k - \hat{x}\|_2^2$  is obtained by using the first inequality of (2.3) since  $f$  is 1-strongly convex.  $\square$

From (4.4), we see that we want to choose  $\alpha$  such that  $L(\alpha)$  is as large as possible:

**Corollary 4.9** *Let Assumption 1 hold true. Then the relaxation parameter  $\alpha$  and the constant  $L$  which yields the fastest convergence rate guarantee in Theorem 4.8 are as follows:*

(a) *General Weights: If  $\eta > \max(1, \sigma_{\max}(\mathbf{W})/2)$  then*

$$\alpha^* = \frac{\|\mathbf{A}\|_F^2}{\sigma_{\max}^2(\mathbf{A})(\eta - 1)} \left( \eta - \frac{\sigma_{\max}(\mathbf{W})}{2} \right),$$

and

$$L(\alpha^*) = \frac{\|\mathbf{A}\|_F^2}{8\sigma_{\max}^2(\mathbf{A})} \cdot \frac{(2\eta - \sigma_{\max}(\mathbf{W}))^2}{\eta(\eta - 1)}.$$

(b) *Uniform Weights, i.e.,  $\mathbf{W} = \alpha\mathbf{I}$ :*

$$\alpha^* = \frac{\eta}{1 + (\eta - 1) \frac{\sigma_{\max}^2(\mathbf{A})}{\|\mathbf{A}\|_F^2}}$$

and

$$L(\alpha^*) = \frac{\eta}{2 + 2(\eta - 1) \frac{\sigma_{\max}^2(\mathbf{A})}{\|\mathbf{A}\|_F^2}}.$$

*Proof* In the case (a) of general weights, in order to get the tightest lower bound, we maximized the concave function  $L(\alpha)$  and obtain

$$\alpha = \|\mathbf{A}\|_F^2(\eta - \sigma_{\max}(\mathbf{W})/2)/((\eta - 1)\sigma_{\max}(\mathbf{A})^2)$$

which fulfills (4.3) and thus gives the best  $q$  is Theorem 4.8. Since we need  $\alpha \geq 0$ , we have to assume  $\eta \geq \sigma_{\max}(\mathbf{W})/2$ . This gives  $\alpha^*$  and plugging this into  $L(\alpha)$  give us  $L(\alpha^*)$ . In the case (b) of uniform weights, we maximized  $L(\alpha)$  for all  $\eta$  with  $\mathbf{W} = \alpha\mathbf{I}$ . □

A few remarks about the interpretation of the above theorem are in order:

*Remark 4.10* (Overrelaxation)

- When a single thread  $\eta = 1$  is used in the case (b) of uniform weight, we see that our optimal relaxation parameter is  $\alpha^* = 1$ . Whereas, when multiple threads  $\eta > 1$  are used, we see

$$1 < \alpha^* \leq \eta,$$

i.e., the method allows for large overrelaxation when the number of threads is high and we will see in Section 5.2 that this does indeed lead to faster convergence.

- Our relaxation parameter  $\alpha^*$  in the case of uniform weights is the same as the relaxation parameter  $\alpha^{RT}$  suggested in [42] although they do not treat the sparse case and only consider uniform weights.
- Finally, for  $\eta = 1$  in the case of general weights, we have, due to the coupling in Assumption 1, that the weights fulfill  $w_i = \frac{\alpha \|a_i\|_2^2}{p_i \|\mathbf{A}\|_F^2}$ . We could estimate  $\sigma_{\max}(\mathbf{W}) \leq \frac{\alpha}{\min_i p_i}$  but this would be a quite crude estimate. By choosing the classical probabilities  $p_i = \|a_i\|_2^2 / \|\mathbf{A}\|_F^2$ , we would get  $\sigma_{\max}(\mathbf{W}) = \alpha$  and get that the relaxation parameter needs to fulfill  $\alpha \in (0, 2)$  and that  $\alpha^* = 1$  is the optimal relaxation parameter.

*Remark 4.11* (Relation to standard randomized sparse Kaczmarz) In the case  $\mathbf{W} = \mathbf{I}$ ,  $\eta = 1$ ,  $\alpha = 1$ , we have  $\mathbf{T} = \frac{1}{2}\mathbf{I}$  and we recover the rate of the standard RSK. It holds

$$\mathbb{E}_k \left[ D_f^{x_k^*}(x_{k+1}, \hat{x}) \right] \leq D_f^{x_k^*}(x_k, \hat{x}) - \frac{1}{2\|\mathbf{A}\|_F^2} \|\mathbf{A}x_k - b\|_2^2$$

which is obtained in [43] and it is shown that this leads to a linear convergence rate in expectation,

$$\mathbb{E} \left[ \|x_k - \hat{x}\|_F^2 \right] \leq 2 \cdot \left( 1 - \frac{1}{2\gamma \|A\|_F^2} \right)^k \cdot f(\hat{x}). \tag{4.5}$$

which implies that we reach accuracy  $\mathbb{E} \left[ \|x_k - \hat{x}\|_2^2 \right] \leq 2 \cdot \varepsilon \cdot f(\hat{x})$  in at most  $k \geq 2\gamma \|A\|_F^2 \log(\frac{1}{\varepsilon})$  iterations.

*Remark 4.12* (Convergence rate for the linearized Bregman method) Similarly to Lemma 4.6, we can show that the linearized Bregman algorithm [3, 23, 48]

$$\begin{aligned} x_{k+1}^* &= x_k^* - \frac{A^T(Ax_k - b)}{\|A\|_2^2}, \\ x_{k+1} &= S_\lambda(x_{k+1}^*) \end{aligned} \tag{4.6}$$

has linear convergence rate given by

$$\|x_k - \hat{x}\|_2^2 \leq 2 \cdot \left( 1 - \frac{1}{2\gamma \|A\|_2^2} \right)^k \cdot f(\hat{x}). \tag{4.7}$$

Although we suspect that this result (4.7) is not new, we could not find it in the literature.

Inspired by [42], the following remarks apply to the uniform weight case.

*Remark 4.13* (Mini-batch vs. full-batch) Let  $H(\eta) = \frac{1}{L(\alpha^*)} = \frac{2}{\eta} + 2(1 - \frac{1}{\eta}) \frac{\sigma_{\max}^2(A)}{\|A\|_F^2}$  be the inverse of  $L(\alpha^*)$ . Recall from Theorem 4.8 that  $L(\alpha^*)$  influences the convergence rate: The larger  $L(\alpha^*)$ , the faster the convergence.

Since  $\frac{\|A\|_F^2}{\sigma_{\max}^2(A)} \geq 1$ ,  $H$  is a nonincreasing function of  $\eta$  and we have  $H(1) = 2$ ,  $H(\infty) \stackrel{\text{def}}{=} \lim_{\eta \rightarrow \infty} H(\eta) = \frac{2\sigma_{\max}^2(A)}{\|A\|_F^2}$ . In the asymptotic regime  $\eta \rightarrow \infty$ , Algorithm 1 becomes linearized Bregman algorithm for minimizing (3.1), and  $H(\infty)$  is the rate of linearized Bregman cf. (4.7). This shows that the averaging method interpolates between the basic method and the linearized Bregman. By increasing  $\eta$ , the quantity  $\frac{H(1)}{H(\infty)} = \frac{\|A\|_F^2}{\sigma_{\max}^2(A)}$  controls the maximum (guaranteed) speedup in the iteration complexity achievable. Comparing (4.4) with the convergence rate (4.5) of the basic sparse Kaczmarz method, we get an improvement of  $2L(\alpha^*) > 1$  which shows that for the RSKA algorithm, we can get a speed-up even of order approximately  $\frac{\|A\|_F^2}{\sigma_{\max}^2(A)}$  compared to the rate of the basic sparse Kaczmarz algorithm (see also the comparison in Table 1).

For  $\eta \geq \frac{\|A\|_F^2}{\sigma_{\max}^2(A)}$ , we get  $H(\eta) \leq 2H(\infty)$ , which is the performance of the full batch (up to a factor of 2). This means that it does not make sense to use a minibatch size larger than  $\frac{\|A\|_F^2}{\sigma_{\max}^2(A)}$ . Moreover, notice that  $H(\eta) \geq \frac{1}{\eta} H(1)$  for all  $\eta$ , show that the

number of iterations does not decrease linearly in the minibatch size  $\eta$ . From a total complexity perspective cf. Table 1, this also means that in a computational regime where processing  $\eta$  basic method updates costs  $\eta$  times as much as processing a single update, the decrease in iteration complexity cannot compensate for the increase in cost per iteration, which means that the choice  $\eta = 1$  is optimal. On the other hand, if a parallel processor is available, a larger  $\eta$  will be better.

### 4.2 Noisy right hand sides

In the noisy case, we consider a consistent linear system  $\mathbf{A}x = b$ , but assume that instead of  $b$  we only have access to some vector  $b^\delta$  with  $|b - b^\delta|_2 \leq \delta$ . In the context of Kaczmarz methods, such errors in the right hand side have been considered in [30, 49]. Since the system  $\mathbf{A}x = b^\delta$  is most likely inconsistent, RSKA will not solve the optimization problem (1.5) but hopefully the iterates still come close to the solution. If we assume a bound  $\|b - b^\delta\|_2 \leq \delta$  for  $b$  with  $\mathbf{A}\hat{x} = b$ , we get the following result on the convergence of the method:

**Theorem 4.14** (Noisy case) *Assume that instead of exact data  $b \in \text{Range}(\mathbf{A})$  only a noisy right hand side  $b^\delta \in \mathbb{R}^m$  with  $\|b^\delta - b\|_2 \leq \delta$  is given. Consider  $\eta > 1$ ,  $\varepsilon > 0$ ,  $\gamma$  as defined in (4.2) (Lemma (4.4)), let Assumption 1 hold and assume that*

$$0 < \alpha < 2 \frac{((1 - \varepsilon)\eta - \frac{1}{2}\sigma_{\max}(\mathbf{W}))\|\mathbf{A}\|_F^2}{\sigma_{\max}(\mathbf{A})^2(\eta - 1)}. \tag{4.8}$$

*If the iterates  $x_k$  of the RSKA method from Algorithm 1 are computed with  $b$  replaced by  $b^\delta$ , then, with the contraction factor  $a$ , we have :*

$$a = 1 - \frac{\alpha}{\gamma} \cdot \frac{(1 - \varepsilon - \sigma_{\max}(\mathbf{T}))}{\|\mathbf{A}\|_F^2} \in (0, 1), \tag{4.9}$$

*and the expected rate of convergence is*

$$\begin{aligned} \mathbb{E} \left[ D_f^{x_k^*}(x_{k+1}, \hat{x}) \right] &\leq a \cdot \mathbb{E} \left[ D_f^{x_k^*}(x_k, \hat{x}) \right] \\ + c\delta^2 \mathbb{E} \left[ \|x_k - \hat{x}\|_2^2 \right] &\leq 2 \cdot a^k \cdot f(\hat{x}) + \frac{c}{1 - a} \delta^2. \end{aligned}$$

where  $c = \frac{\alpha}{\|\mathbf{A}\|_F^2} \left( \sigma_{\max}(\mathbf{T}) + \frac{1}{\varepsilon} \sigma_{\max}^2(\mathbf{T}') \right)$ ,  $\mathbf{T}' = \mathbf{T} - \frac{1}{2}\mathbf{I}$

**Table 1** Complexity of different methods

	RSK	RSKA	linBreg
Iteration complexity	$\mathcal{O}\left(2\gamma\ \mathbf{A}\ _F^2 \log\left(\frac{1}{\varepsilon}\right)\right)$	$\mathcal{O}\left(\gamma \frac{\ \mathbf{A}\ _F^2}{L(\alpha^\gamma)} \log\left(\frac{1}{\varepsilon}\right)\right)$	$\mathcal{O}\left(2\gamma\ \mathbf{A}\ _2^2 \log\left(\frac{1}{\varepsilon}\right)\right)$
Cost per iteration	$\mathcal{O}(n)$	$\mathcal{O}(\eta n)$	$\mathcal{O}(mn)$

*Proof* Assuming that a noisy observed data  $b^\delta \in \mathbb{R}^m$  instead of  $b$  with  $\|b^\delta - b\|_2 \leq \delta$  is given, where  $b = A\hat{x}$ . The update in this case is given by:

$$\begin{aligned} x_{k+1}^* &= x_k^* - \frac{1}{\eta} \sum_{i \in \tau_k} w_i \frac{\langle a_i, x_k \rangle - b_i^\delta}{\|a_i\|_2^2} \cdot a_i, \\ x_{k+1} &= S_\lambda(x_{k+1}^*) \end{aligned} \tag{4.10}$$

where  $\eta = |\tau_k|$ , which in terms of matrix multiplication is equal to:

$$\begin{aligned} x_{k+1}^* &= x_k^* - \mathbf{A}^T \cdot \mathbf{M}_k \cdot (\mathbf{A}x_k - b^\delta), \\ x_{k+1} &= S_\lambda(x_{k+1}^*) \end{aligned} \tag{4.11}$$

We introduce the abbreviation

$$x_k^\delta \stackrel{\text{def}}{=} \hat{x} + \mathbf{A}^T \cdot \mathbf{M}_k \cdot (b - b^\delta)$$

and use Lemma 4.3 with  $f(x) = \lambda\|x\|_1 + \frac{1}{2}\|x\|_2^2$  and  $\Phi(x) = \langle \mathbf{A}^T \mathbf{M}_k (\mathbf{A}x_k - b^\delta), x - x_k \rangle$ , and  $y = x_k^\delta$  and get

$$\begin{aligned} D_f^{x_{k+1}^*}(x_{k+1}, x_k^\delta) &\leq D_f^{x_k^*}(x_k, x_k^\delta) + \Phi(x_k^\delta) - \Phi(x_{k+1}) - D_f^{x_k^*}(x_k, x_{k+1}) \\ &= D_f^{x_k^*}(x_k, x_k^\delta) - \langle \mathbf{A}^T \mathbf{M}_k (\mathbf{A}x_k - b^\delta), x_k - x_k^\delta \rangle \\ &\quad + \langle \mathbf{A}^T \mathbf{M}_k (\mathbf{A}x_k - b^\delta), x_k - x_{k+1} \rangle - D_f^{x_k^*}(x_k, x_{k+1}) \\ &\leq D_f^{x_k^*}(x_k, x_k^\delta) - \langle \mathbf{A}^T \mathbf{M}_k (\mathbf{A}x_k - b^\delta), x_k - x_k^\delta \rangle \\ &\quad + \|\mathbf{A}^T \mathbf{M}_k (\mathbf{A}x_k - b^\delta)\| \cdot \|x_k - x_{k+1}\| - \frac{1}{2}\|x_k - x_{k+1}\|^2 \\ &\leq D_f^{x_k^*}(x_k, x_k^\delta) - \langle \mathbf{A}^T \mathbf{M}_k (\mathbf{A}x_k - b^\delta), x_k - x_k^\delta \rangle \\ &\quad + \frac{1}{2}\|\mathbf{A}^T \mathbf{M}_k (\mathbf{A}x_k - b^\delta)\|^2 \end{aligned}$$

so that

$$D_f^{x_{k+1}^*}(x_{k+1}, x_k^\delta) \leq D_f^{x_k^*}(x_k, x_k^\delta) - \langle \mathbf{A}^T \mathbf{M}_k (\mathbf{A}x_k - b^\delta), x_k - x_k^\delta \rangle + \frac{1}{2}\|\mathbf{A}^T \mathbf{M}_k (\mathbf{A}x_k - b^\delta)\|^2 \tag{4.12}$$

Unfolding the expression of  $D_f^{x_{k+1}^*}(x_{k+1}, x_k^\delta)$  and  $D_f^{x_k^*}(x_k, x_k^\delta)$ , we get:

$$D_f^{x_{k+1}^*}(x_{k+1}, \hat{x}) \leq D_f^{x_k^*}(x_k, \hat{x}) - \langle \mathbf{A}^T \mathbf{M}_k (\mathbf{A}x_k - b^\delta), x_k - \hat{x} \rangle + \frac{1}{2}\|\mathbf{A}^T \mathbf{M}_k (\mathbf{A}x_k - b^\delta)\|_2^2$$

On the other hand,

$$\mathbb{E}_k [\langle \mathbf{M}_k \cdot (\mathbf{A}x_k - b^\delta), \mathbf{A}x_k - b \rangle] = \frac{\alpha}{\|\mathbf{A}\|_F^2} \|\mathbf{A}x_k - b\|_2^2 + \frac{\alpha}{\|\mathbf{A}\|_F^2} \langle b - b^\delta, \mathbf{A}x_k - b \rangle$$

and

$$\begin{aligned} \mathbb{E}_k \left[ \left\| \mathbf{A}^T \cdot \mathbf{M}_k \cdot (\mathbf{A}x_k - b^\delta) \right\|_2^2 \right] &= \langle \mathbf{A}x_k - b, \mathbb{E}_k \left[ \mathbf{M}_k^T \mathbf{A} \mathbf{A}^T \mathbf{M}_k \right] \cdot (\mathbf{A}x_k - b) \rangle \\ &\quad + \langle b - b^\delta, \mathbb{E}_k \left[ \mathbf{M}_k^T \mathbf{A} \mathbf{A}^T \mathbf{M}_k \right] \cdot (b - b^\delta) \rangle \\ &\quad + 2 \langle \mathbf{A}x_k - b, \mathbb{E}_k \left[ \mathbf{M}_k^T \mathbf{A} \mathbf{A}^T \mathbf{M}_k \right] \cdot (b - b^\delta) \rangle \end{aligned}$$

In summary, we have:

$$\begin{aligned}
 & - \mathbb{E}_k \left[ \langle \mathbf{A}^T \cdot \mathbf{M}_k \cdot (\mathbf{A}x_k - b^\delta), x_k - \hat{x} \rangle \right] + \frac{1}{2} \mathbb{E}_k \left[ \left\| \mathbf{A}^T \cdot \mathbf{M}_k \cdot (\mathbf{A}x_k - b^\delta) \right\|_2^2 \right] \\
 &= - \frac{\alpha}{\|\mathbf{A}\|_F^2} \|\mathbf{A}x_k - b\|_2^2 - \frac{\alpha}{\|\mathbf{A}\|_F^2} \langle b - b^\delta, \mathbf{A}x_k - b \rangle \\
 & \quad + \frac{1}{2} \langle b - b^\delta, \mathbb{E}_k \left[ \mathbf{M}_k^T \mathbf{A} \mathbf{A}^T \mathbf{M}_k \right] (b - b^\delta) \rangle \\
 & \quad + \frac{1}{2} \langle \mathbf{A}x_k - b, \mathbb{E}_k \left[ \mathbf{M}_k^T \mathbf{A} \mathbf{A}^T \mathbf{M}_k \right] (\mathbf{A}x_k - b) \rangle \\
 & \quad + \langle \mathbf{A}x_k - b, \mathbb{E}_k \left[ \mathbf{M}_k^T \mathbf{A} \mathbf{A}^T \mathbf{M}_k \right] \cdot (b - b^\delta) \rangle \\
 & \leq - \frac{\alpha}{\|\mathbf{A}\|_F^2} \|\mathbf{A}x_k - b\|_2^2 + \frac{\alpha}{\|\mathbf{A}\|_F^2} \sigma_{\max}(\mathbf{T}) \|\mathbf{A}x_k - b\|_2^2 + \frac{\alpha}{\|\mathbf{A}\|_F^2} \sigma_{\max}(\mathbf{T}) \|b - b^\delta\|_2^2 \\
 & \quad + \frac{2\alpha}{\|\mathbf{A}\|_F^2} \langle \mathbf{A}x_k - b, \left( \frac{1}{2\eta} \mathbf{W} - \frac{1}{2} I + \frac{\alpha}{2} \left( 1 - \frac{1}{\eta} \right) \frac{\mathbf{A} \mathbf{A}^T}{\|\mathbf{A}\|_F^2} \right) \cdot (b - b^\delta) \rangle \\
 & \leq - \frac{\alpha}{\|\mathbf{A}\|_F^2} \|\mathbf{A}x_k - b\|_2^2 + \frac{\alpha}{\|\mathbf{A}\|_F^2} \sigma_{\max}(\mathbf{T}) \|\mathbf{A}x_k - b\|_2^2 + \frac{\alpha}{\|\mathbf{A}\|_F^2} \sigma_{\max}(\mathbf{T}) \|b - b^\delta\|_2^2 \\
 & \quad + \frac{\alpha \varepsilon}{\|\mathbf{A}\|_F^2} \|\mathbf{A}x_k - b\|_2^2 + \frac{\alpha}{\varepsilon \|\mathbf{A}\|_F^2} \left\| \left( \frac{1}{2\eta} \mathbf{W} - \frac{1}{2} I + \frac{\alpha}{2} \left( 1 - \frac{1}{\eta} \right) \frac{\mathbf{A} \mathbf{A}^T}{\|\mathbf{A}\|_F^2} \right) \cdot (b - b^\delta) \right\|_2^2 \\
 &= - \frac{\alpha}{\|\mathbf{A}\|_F^2} (1 - \varepsilon - \sigma_{\max}(\mathbf{T})) \|\mathbf{A}x_k - b\|_2^2 \\
 & \quad + \frac{\alpha}{\|\mathbf{A}\|_F^2} (\sigma_{\max}(\mathbf{T}) + \frac{1}{\varepsilon} \sigma_{\max}^2(\mathbf{T}')) \|b - b^\delta\|_2^2. \tag{4.13}
 \end{aligned}$$

To get an improvement after each iteration, we need  $(1 - \varepsilon - \sigma_{\max}(\mathbf{T})) > 0$ , i.e.,  $\sigma_{\max}(\mathbf{T}) < 1 - \varepsilon$  which hold true because of (4.8). Combining (4.12) and (4.13) and using the error bound from Lemma 4.4, we arrive at

$$\begin{aligned}
 \mathbb{E}_k \left[ D_f^{x_k^*+1}(x_{k+1}, \hat{x}) \right] & \leq \left( 1 - \frac{\alpha}{\gamma \|\mathbf{A}\|_F^2} (1 - \varepsilon - \sigma_{\max}(\mathbf{T})) \right) D_f^{x_k^*}(x_k, \hat{x}) \\
 & \quad + \frac{\alpha}{\|\mathbf{A}\|_F^2} (\sigma_{\max}(\mathbf{T}) + \frac{1}{\varepsilon} \sigma_{\max}^2(\mathbf{T}')) \delta^2,
 \end{aligned}$$

which concludes the proof. The inequality in terms of the norm is obtained by using the first inequality of (2.3) since  $f$  is 1-strongly convex. □

*Remark 4.15* Note that for  $\mathbf{W} = \mathbf{I}$ ,  $\eta = 1$ ,  $\alpha = 1$ , we have  $\mathbf{T} = \frac{1}{2} \mathbf{I}$  meaning  $\mathbf{T}' = 0$  and sending  $\varepsilon$  to zero give us  $\frac{c}{1-\alpha} = \gamma$  which recover the rate of the standard RSK in the noisy case showed in [43].

### 5 Numerical experiments

We present several experiments to demonstrate the effectiveness of Algorithm 1 under various conditions. In particular, we study the effects of the relaxation parameter  $\alpha$ , the number of threads  $\eta$ , the sparsity parameter  $\lambda$ , the weight matrix  $\mathbf{W}$ , and the probability matrix  $\mathbf{P}$ . The simulations were performed in Python on an Intel Core i7 computer with 16GB RAM. We start by comparing several variants of RSKA algorithms with randomized Kaczmarz (RK) and randomized sparse Kaczmarz (RSK). We consider the following RSKA variants:

- (a) v1: RSKA with  $\mathbf{W} = \mathbf{I}$ , i.e.,  $\alpha = 1$ , with a coupling such that  $\mathbf{PWD}^{-2} = \frac{\alpha \mathbf{I}}{\|\mathbf{A}\|_F^2}$   
 i.e.,  $p_i = \frac{\|a_i\|_2^2}{\|\mathbf{A}\|_F^2}$ .
- (b) v2: RSKA with a uniform weight matrix  $\mathbf{W} = \alpha^* \mathbf{I}$ , i.e.,  $p_i = \frac{\|a_i\|_2^2}{\|\mathbf{A}\|_F^2}$ , where  $\alpha^*$  is from Corollary 4.9.
- (c) v3: RSKA with a general diagonal weight matrix  $\mathbf{W}$  with diagonal entries  $w_i$  sample i.i.d. from the uniform distribution on  $[0, 1]$  and  $p_i = \frac{\|a_i\|_2^2}{\|\mathbf{A}\|_F^2}$ .
- (d) v4: RSKA with a general diagonal weight matrix  $\mathbf{W}$  with diagonal entries  $w_i$  sample i.i.d. from the uniform distribution on  $[0, 1]$  and set  $p_i$  proportional to  $\frac{\|a_i\|_2^2}{w_i}$ , i.e., we have  $\mathbf{PWD}^{-2} = \alpha \mathbf{I}$  with  $\alpha = (\sum_j \frac{\|a_j\|_2^2}{w_j})^{-1}$ .

The rationale behind these choices are as follows: Version v1 just chooses no weight and standard probabilities. In v2, we still choose standard probabilities but use a coupling of weights and probabilities with the uniform weight  $\alpha^*$  of which we have seen in Corollary 4.9 that is has a certain optimality property. In v3, we still use standard probabilities but do not use a coupling of probabilities and weights. Since we do not have any indications on how to choose weights in any optimal way, we just used random weights here. In contrast to v3, we do use a coupling of weights and probabilities in v4, but again, since we do not have results on how to choose optimal weights, we choose random ones.

Note that RSK and RK are both special cases of RSKA, namely

- (a) RSKA with  $\mathbf{W} = \mathbf{I}$ , i.e.,  $\alpha = 1$ , with a coupling such that  $\mathbf{PWD}^{-2} = \frac{\alpha \mathbf{I}}{\|\mathbf{A}\|_F^2}$ , i.e.,  
 $p_i = \frac{\|a_i\|_2^2}{\|\mathbf{A}\|_F^2}$  and  $\eta = 1$ ,
- (b) RSKA with  $\mathbf{W} = \mathbf{I}$ , i.e.,  $\alpha = 1$ , with a coupling such that  $\mathbf{PWD}^{-2} = \frac{\alpha \mathbf{I}}{\|\mathbf{A}\|_F^2}$ , i.e.,  
 $p_i = \frac{\|a_i\|_2^2}{\|\mathbf{A}\|_F^2}$ ,  $\eta = 1$  and  $\lambda = 0$ .

Synthetic data for the experiments is generated as follows: All elements of the data matrix  $\mathbf{A} \in \mathbb{R}^{m \times n}$  are chosen independent and identically distributed from the standard normal distribution  $\mathcal{N}(0, 1)$ . We constructed overdetermined, square, and underdetermined linear systems. To construct sparse solutions  $\hat{x} \in \mathbb{R}^n$ , we choose  $s$  indices from  $\{1, \dots, n\}$  at random and placed zeros at these positions, and the corresponding right hand sides are  $b = \mathbf{A}\hat{x} \in \mathbb{R}^m$  while the respective noisy right hand

sides are  $b^\delta$  and are obtained by adding Gaussian noise (see Section 5.4 below). We generally chose  $\eta = 1 + \frac{1}{10} \min(m, n)$  for RSKA, unless something else is indicated. Note that in the overdetermined case with no noise, there will be a unique solution  $\hat{x}$  since with probability 1, the matrices  $\mathbf{A}$  have full rank, and so all methods are expected to converge to the same solution  $\hat{x}$  in this case.

For each experiment, we run independent trials each starting with the initial iterate  $x_0 = 0$ . We measure performance by plotting the relative residual error  $\|Ax - b\|/\|b\|$  and the error  $\|x_k - \hat{x}\|/\|\hat{x}\|$  against the number of full iterations. Thick line shows mean over the total number of trials and shaded area which represent the standard deviation over the trials are plotted when appropriate.

Figure 1 shows the result for a five times overdetermined and consistent system without noise where the value  $\lambda = 1$  was used for RSK and RSKA. Note that the usual RK and RSKA variants perform consistently well over all trials, while the performance of RSK differs drastically between different instances. Moreover, we observe experimentally that choosing the theoretically optimal overrelaxation parameter  $\alpha^*$  from Corollary 4.9 for the RSKA v2 method gives us faster convergence.

Figures 2 and 3 show the results respectively for a two times resp. five times underdetermined and consistent system without noise where the values  $\lambda = 1$ , resp.  $\lambda = 3$  was used for RSK and RSKA. Methods like RSK and RSKA take advantage of the fact that the vectors  $\hat{x}$  are very sparse. Moreover, in Fig. 2, the RSK and RK methods do not reduce the residual as fast as the RSKA method. However, since the problem is underdetermined, the RK method does not converge to a sparse solution

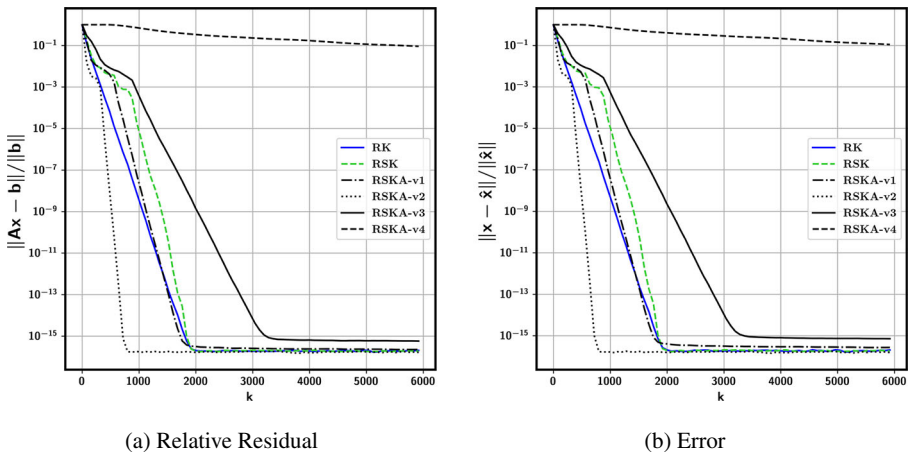
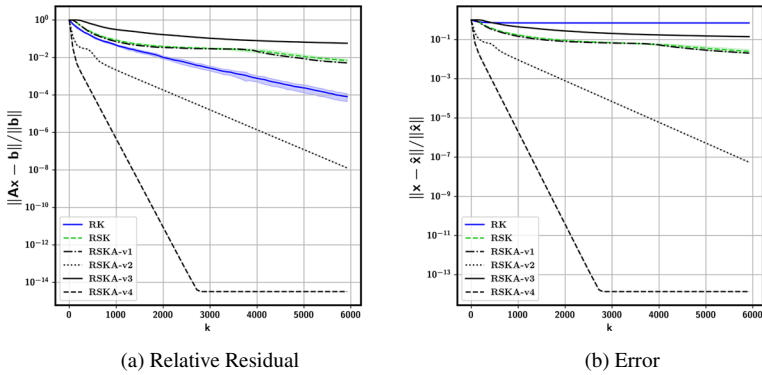


Fig. 1 A comparison of randomized Kaczmarz (blue), randomized sparse Kaczmarz (green), and RSKA method (black),  $m = 100, n = 20$ , sparsity  $s = 10, \eta = 11, \lambda = 1$ , no noise, and 20 runs. Thick line shows mean over all trials



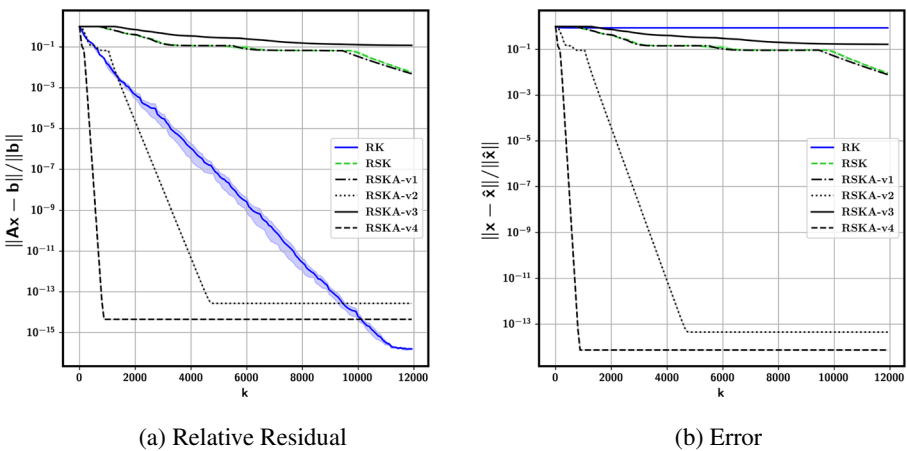


**Fig. 2** A comparison of randomized Kaczmarz (blue), randomized sparse Kaczmarz (green), and RSKA method (black),  $m = 100, n = 200$ , sparsity  $s = 10, \eta = 11, \lambda = 1$ , no noise, and 10 runs. Thick line shows mean over all trials, and shaded area represents the standard deviation

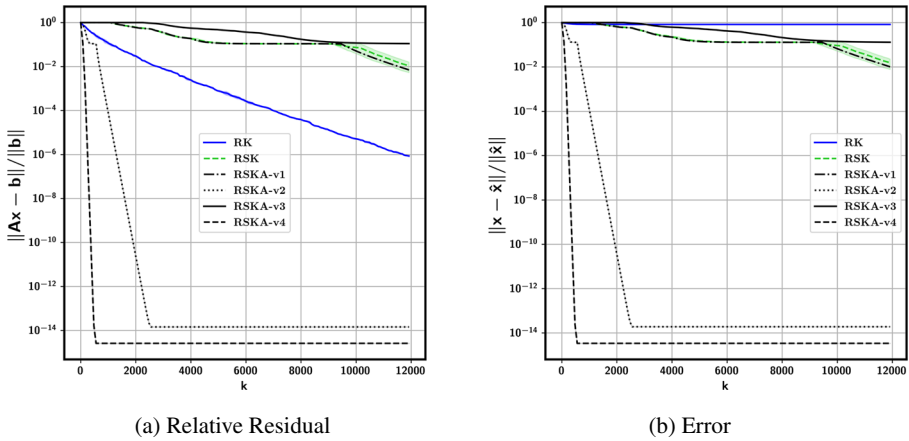
and hence, the error does not converge to zero. Figures 4 and 5 show results for further values of  $m$  and  $n$  and show similar behavior as Fig. 2.

### 5.1 The effect of the number of threads $\eta$

In Figs. 6, 7, 8, and 9, we see the effects of the number of threads  $\eta$  in the error of Algorithm 1 for the variant v2. We used underdetermined and overdetermined and consistent system, with  $\lambda \in \{0.01, 3\}$ . For the small  $\lambda = 0.01$ , the RSKA

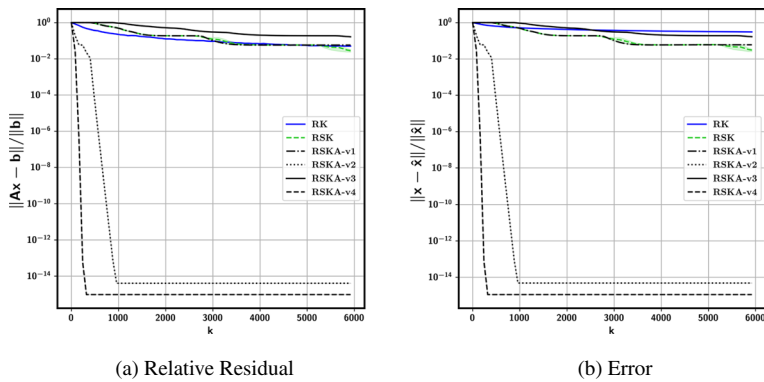


**Fig. 3** A comparison of randomized Kaczmarz (blue), randomized sparse Kaczmarz (green), and RSKA method (black),  $m = 100, n = 500$ , sparsity  $s = 10, \eta = 11, \lambda = 3$ , no noise, and 10 runs. Thick line shows mean over all trials, and shaded area represents the standard deviation

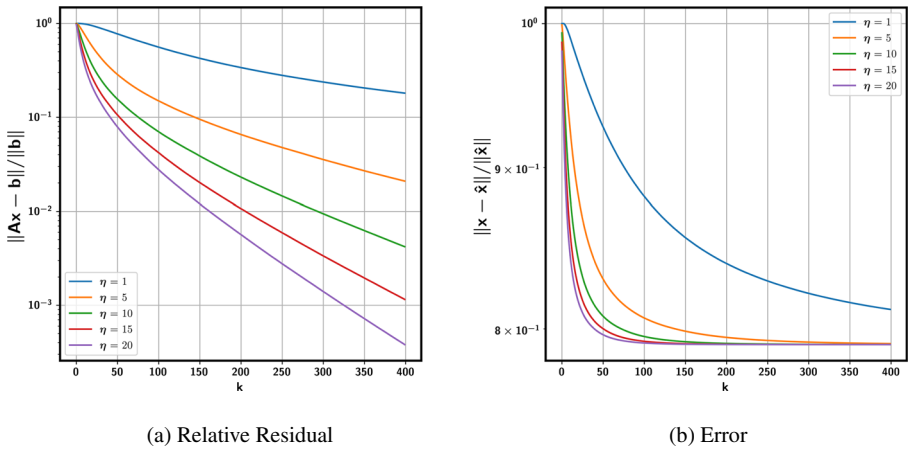


**Fig. 4** A comparison of randomized Kaczmarz (blue), randomized sparse Kaczmarz (green), and RSKA method (black),  $m = 200$ ,  $n = 600$ , sparsity  $s = 10$ ,  $\eta = 21$ ,  $\lambda = 3$ , no noise, and 5 runs. Thick line shows mean over all trials, and shaded area represents the standard deviation

behaves almost like the standard randomized Kaczmarz with averaging from [28], while for the larger value  $\lambda = 3$ , we see the typical behavior for the sparse Kaczmarz method which stagnates from time to time and switches to faster improvement in between [23]. As the number of threads  $\eta$  increases, we see a corresponding decrease in the number of iterations needed to reach a certain accuracy; however, at some point, increasing  $\eta$  does not improve the method in accordance with Remark 4.10. For smaller values of  $\eta$ , we roughly see an  $\eta$ -times speedup in the number of iterations. Thus, it is clear that the averaging will pay off as soon as the updates in RSKA can be done in parallel.



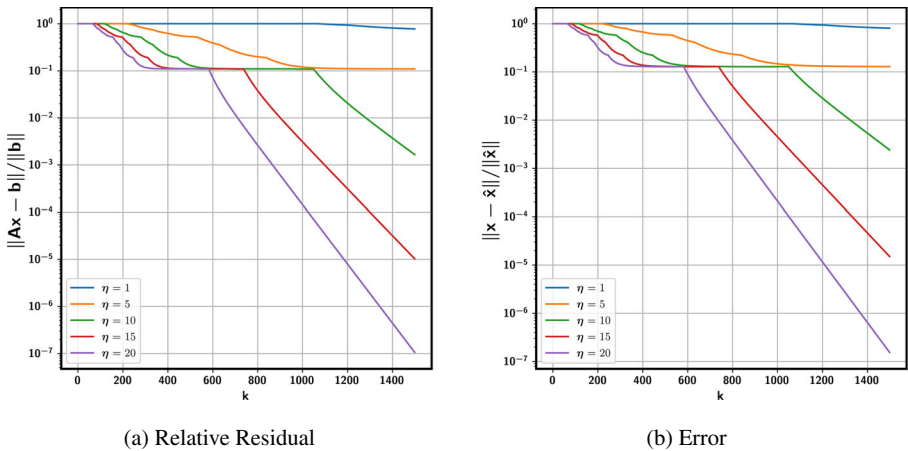
**Fig. 5** A comparison of randomized Kaczmarz (blue), randomized sparse Kaczmarz (green), and RSKA method (black),  $m = 300$ ,  $n = 300$ , sparsity  $s = 10$ ,  $\eta = 31$ ,  $\lambda = 3$ , no noise, and 5 runs. Thick line shows mean over all trials, and shaded area represents the standard deviation



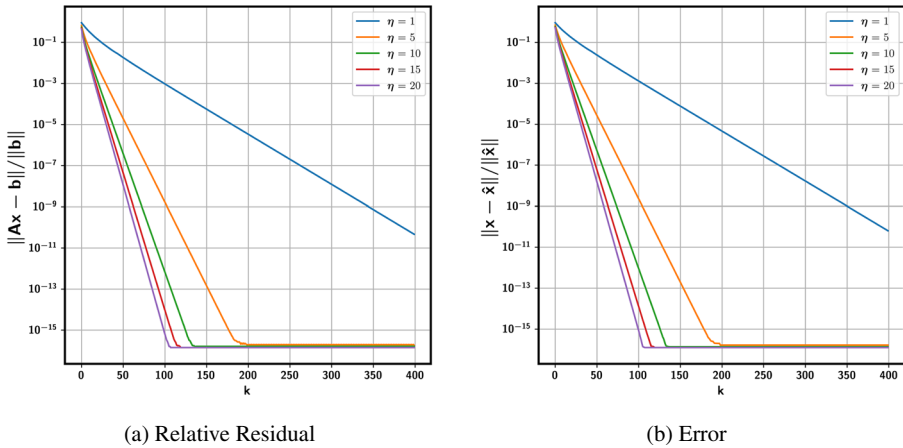
**Fig. 6** The effect of the number of threads  $\eta$  on the error and relative residual versus iteration for Algorithm 1 on the variant  $v2$ .  $m = 200$ ,  $n = 600$ , sparsity  $s = 10$ ,  $\lambda = 0.01$ , no noise

### 5.2 The effect of the relaxation parameter $\alpha$

In Fig. 10, we observe the effect on the convergence rate as we vary the relaxation parameter  $\alpha$ . We used an underdetermined and consistent system with  $\lambda = 0.1$ ,  $\eta = 8$  with variant  $v2$ . In fact, increasing  $\alpha$  allows us to get smaller error; however, the method can ultimately diverge for some larger values of the relaxation parameter  $\alpha$ . This is observed in Fig. 11 which plot the relative residual and the error after 100

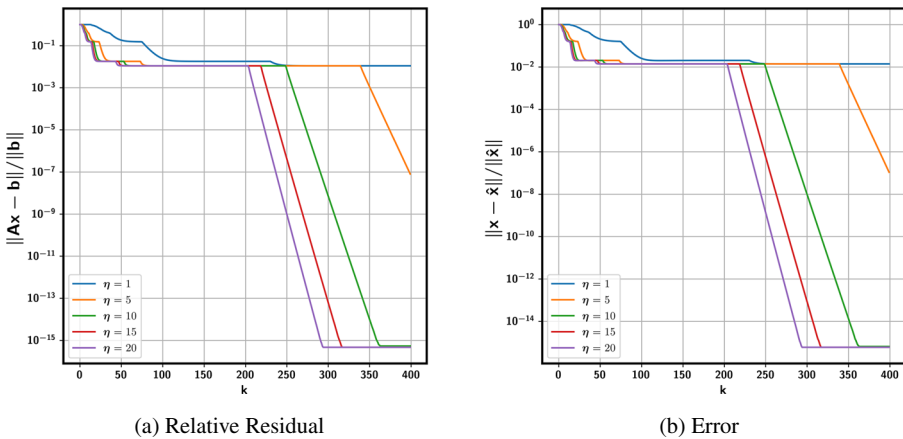


**Fig. 7** The effect of the number of threads  $\eta$  on the error and relative residual versus iteration for Algorithm 1 on the variant  $v2$ .  $m = 200$ ,  $n = 600$ , sparsity  $s = 10$ ,  $\lambda = 3$ , no noise

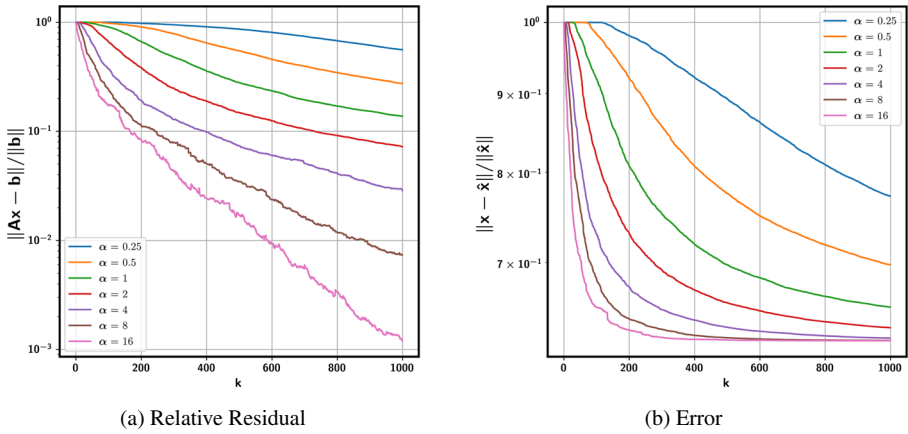


**Fig. 8** The effect of the number of threads  $\eta$  on the error and relative residual versus iteration for Algorithm 1 on the variant  $v2$ .  $m = 100$ ,  $n = 10$ , sparsity  $s = 10$ ,  $\lambda = 0.01$ , no noise

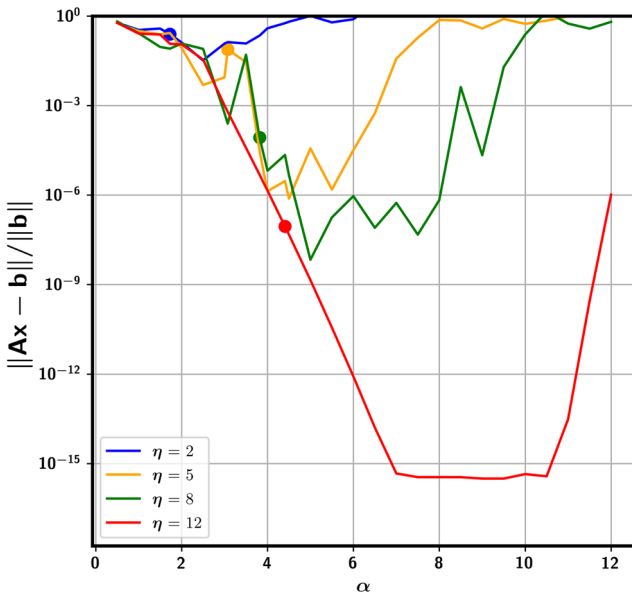
iterations on a smaller example for various relaxation parameters  $\alpha$  and batch sizes  $\eta$ . The theoretically optimal parameter  $\alpha^*$  from Corollary 4.9 is indicated as a dot. We used an overdetermined and consistent system with  $\lambda = 1$  with variant  $v2$ . The plots confirm that  $\alpha$  cannot be chosen too large, i.e., there exists an  $\eta$ -dependent upper bound for  $\alpha$  which leads to convergence (cf. Theorem 4.8). However, we also observe that a larger relaxation parameter than  $\alpha^*$  from Corollary 4.9 leads to even faster convergence.



**Fig. 9** The effect of the number of threads  $\eta$  on the error and relative residual versus iteration for Algorithm 1 on the variant  $v2$ .  $m = 100$ ,  $n = 10$ , sparsity  $s = 10$ ,  $\lambda = 3$ , no noise



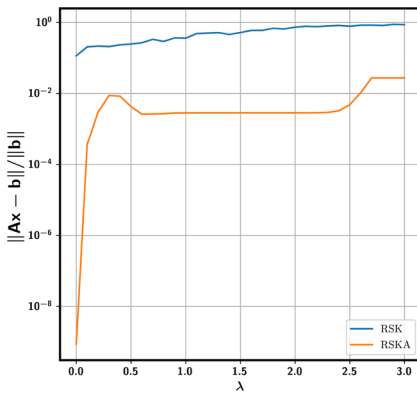
**Fig. 10** The effect of the relaxation parameter  $\alpha$  on the error and relative residual versus iteration for Algorithm 1 on the variant  $v2$ .  $m = 200, n = 600$ , sparsity  $s = 10, \eta = 8, \lambda = 0.1$ , no noise



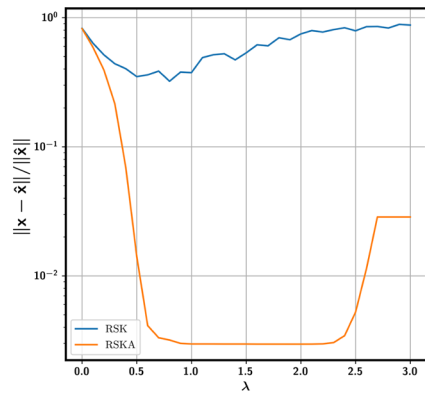
**Fig. 11** The effect of the relaxation parameter  $\alpha$  for various values of  $\eta$  on the relative residual after 100 iterations of Algorithm 1 with the variant  $v2$ .  $m = 100, n = 10$ , sparsity  $s = 10, \lambda = 6$ , no noise. Circle markers are estimates of the optimal relaxation parameter using Corollary 4.9

### 5.3 The effect of the sparsity parameter $\lambda$

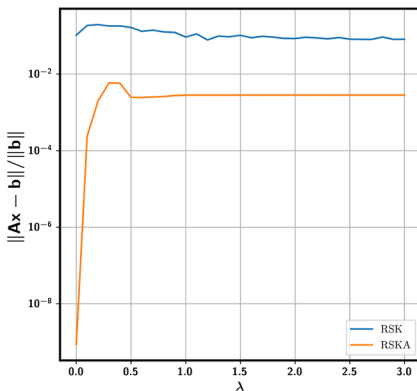
In Fig. 12, we see the effects of the sparsity parameter  $\lambda$  on the approximation error of Algorithm 1 and the randomized Kaczmarz method (RK). We used an underdetermined and consistent system with  $\eta = 21$  with variant *v2* of RSKA. We observed that as you increase the sparsity parameter  $\lambda$ , the relative residual and the reconstruction error get worse and RSK is more affected by this behavior whereas RSKA keep its performance along different  $\lambda$ . The first row of Fig. 12 corresponds to experiment where we run 1000 iterations and in the second row, the methods are run for  $(1 + \lambda)1000$  iterations for all  $\lambda$ .



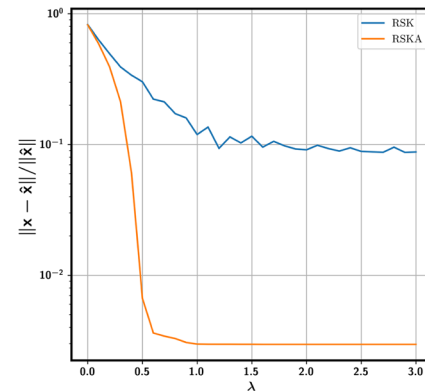
(a) Relative Residual



(b) Error

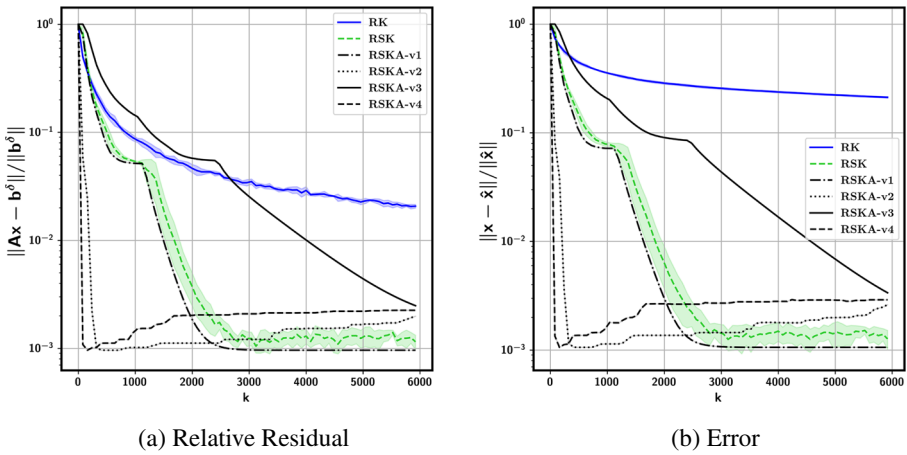


(c) Relative Residual



(d) Error

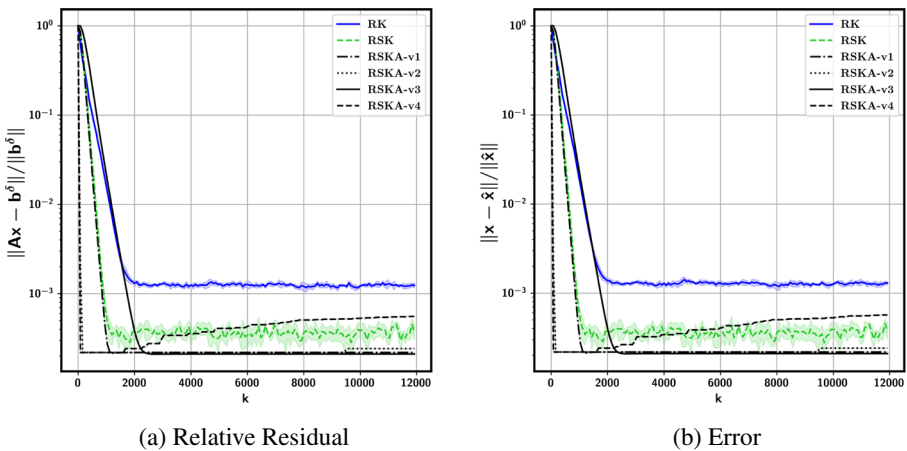
**Fig. 12** A comparison of randomized sparse Kaczmarz (blue) and RSKA-*v2* method (orange) in terms of the sparsity parameter  $\lambda$ .  $m = 200$ ,  $n = 600$ , sparsity  $s = 10$ ,  $\eta = 21$ , no noise. In the first row (**a**, **b**), methods are run for 1000 iterations for all  $\lambda$  whereas in the second row (**c**, **d**), methods are run for  $(1 + \lambda)1000$  iterations for all  $\lambda$



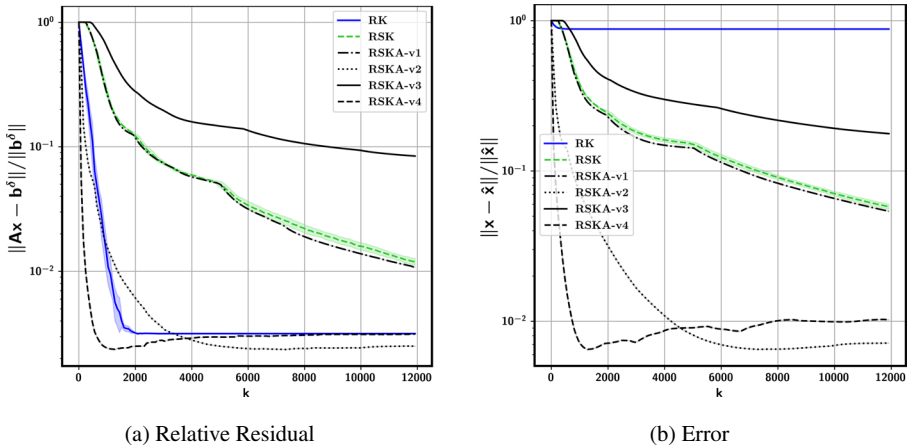
**Fig. 13** A comparison of randomized Kaczmarz (blue), randomized sparse Kaczmarz (green), and RSKA method (black),  $m = 100, n = 100$ , sparsity  $s = 10, \eta = 11, \lambda = 1$ , noise level  $l = 0.1$ , and 5 runs. Thick line shows mean over all trials, and shaded area represents the standard deviation

### 5.4 Noisy case

In this part, we are interested in the effectiveness of the RSKA method on inconsistent systems. We construct a sparse  $\hat{x}$  with normally distributed non-zero entries and set  $b = A\hat{x}, b^\epsilon = b + \epsilon$  where  $\epsilon$  is a random vector uniformly distributed on a sphere with radius  $l$  that correspond to the relative noise level such that  $\|b - b^\epsilon\|_2 \leq l$ .



**Fig. 14** A comparison of randomized Kaczmarz (blue), randomized sparse Kaczmarz (green), and RSKA method (black),  $m = 500, n = 100$ , sparsity  $s = 10, \eta = 11, \lambda = 1$ , noise level  $l = 0.1$ , and 5 runs. Thick line shows mean over all trials, and shaded area represents the standard deviation



**Fig. 15** A comparison of randomized Kaczmarz (blue), randomized sparse Kaczmarz (green), and RSKA method (black),  $m = 100, n = 500$ , sparsity  $s = 10, \eta = 11, \lambda = 1$ , noise level  $l = 0.1$ , and 5 runs. Thick line shows mean over all trials, and shaded area represents the standard deviation

Figures 13, 14, and 15 show the results for noisy right hand sides, all with  $\lambda = 1$  for RSK and RSKA. Figure 14 uses a five times overdetermined system with 10% relative noise, and Fig. 15 has the same noise level and a five times underdetermined system. In the underdetermined case, all methods consistently stagnate at a residual level which is comparable to the noise level; however, in all settings, RSKA variants achieve faster convergence than RSK which in turn is faster than RK. Regarding the reconstruction error, RSKA and RSK achieve reconstructions with an error in the size of the noise level, while RSKA achieves an even lower reconstruction error.

### 6 Conclusion

We proved that the iterates of the randomized sparse Kaczmarz with averaging method (Algorithm 1) are expected to converge linearly for consistent linear systems. Moreover, we show that the iterates reach an error threshold in the order of the noise-level in the noisy case. We gave a general error bound in terms of the sparsity parameter  $\lambda$ , the number of threads  $\eta$ , and a relaxation parameter  $\alpha$ . Numerical experiments show that the method performs consistently well over a range of values of  $\lambda$  (which is different for the version without averaging), very good reconstruction quality as  $\lambda$  increases, confirm the theoretical results, and demonstrate the benefit of using Algorithm 1 to recover sparse solutions of linear systems, even in the noisy case. We demonstrate that the rate of convergence for Algorithm 1 improves both in theory and practice as the number of threads  $\eta$  increases. Moreover, we derive an optimal value for the relaxation parameter  $\alpha$  which gives the fastest convergence speed and our numerical experiments indicate that this optimal value for  $\alpha$  (in v2 and v4 of the algorithms in Section 5) does indeed provide fast convergence.



**Funding** Open Access funding enabled and organized by Projekt DEAL. The work of the authors has been supported by the ITN-ETN project TraDE-OPT funded by the European Union’s Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 861137.

**Availability of data and materials** The data for the numerical example are randomly generated matrices.

**Code availability** The computational code is only prototypal, but it is available from the authors upon request.

## Declarations

**Conflict of interest** The authors declare no competing interests.

**Disclaimer** This work represents only the author’s view and the European Commission is not responsible for any use that may be made of the information it contains.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

## References


1. Beck, A., Teboulle, M.: Mirror descent and nonlinear projected subgradient methods for convex optimization. *Oper. Res. Lett.* **31**(3), 167–175 (2003)
2. Cai, J.F., Osher, S., Shen, Z.: Convergence of the linearized Bregman iteration for  $\ell_1$ -norm minimization. *Math. Comput.* **78**(268), 2127–2136 (2009)
3. Cai, J.F., Osher, S., Shen, Z.: Linearized Bregman iterations for compressed sensing. *Math. Comput.* **78**(267), 1515–1536 (2009)
4. Candès, E.J.: Compressive sampling. In: *International Congress of Mathematicians*, vol. III, pp. 1433–1452. Eur. Math. Soc. Zürich (2006)
5. Candès, E.J., Romberg, J., Tao, T.: Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inf. Theory* **52**(2), 489–509 (2006)
6. Chen, S.S., Donoho, D.L., Saunders, M.A.: Atomic decomposition by basis pursuit. *SIAM J. Sci. Comput.* **20**(1), 33–61 (1998). <https://doi.org/10.1137/S1064827596304010>
7. Donoho, D.L.: Compressed sensing. *IEEE Trans. Inform. Theory* **52**(4), 1289–1306 (2006). <https://doi.org/10.1109/TIT.2006.871582>
8. Donoho, D.L., Tanner, J.: Sparse nonnegative solution of underdetermined linear equations by linear programming. *Proc. Natl. Acad. Sci.* **102**(27), 9446–9451 (2005)
9. D’Orazio, R., Loizou, N., Laradji, I., Mitliagkas, I. (2021)
10. Du, K., Si, W.T., Sun, X.H.: Randomized extended average block Kaczmarz for solving least squares. *SIAM J. Sci. Comput.* **42**(6), A3541–A3559 (2020)
11. Friedlander, M.P., Tseng, P.: Exact regularization of convex programs. *SIAM J. Optim.* **18**(4), 1326–1350 (2008)
12. Garey, M.R., Johnson, D.S.: *Computers and Intractability. A Series of Books in the Mathematical Sciences.* W. H. Freeman and Co., San Francisco, Calif. (1979). A guide to the theory of NP-completeness
13. Gower, R.M., Molitor, D., Moorman, J., Needell, D.: On adaptive sketch-and-project for solving linear systems. *SIAM J. Matrix Anal. Appl.* **42**(2), 954–989 (2021). <https://doi.org/10.1137/19M1285846>

14. Hanke, M., Niethammer, W.: On the acceleration of Kaczmarz's method for inconsistent linear systems. *Linear Algebra Appl.* **130**, 83–98 (1990)
15. Herman, G.T., Lent, A., Lutz, P.H.: Relaxation methods for image reconstruction. *Commun. ACM* **21**(2), 152–158 (1978). <https://doi.org/10.1145/359340.359351>
16. Horn, R.A., Horn, R.A., Johnson, C.R.: *Topics in matrix analysis*. Cambridge University Press, Cambridge (1994)
17. Hounsfield, G.N.: Computerized transverse axial scanning (tomography): part 1. description of system. *Br. J. Radiol.* **46**(552), 1016–1022 (1973)
18. Jiao, Y., Jin, B., Lu, X.: Preasymptotic convergence of randomized Kaczmarz method. *Inverse Prob.* **33**(12), 125,012, 21 (2017). <https://doi.org/10.1088/1361-6420/aa8e82>
19. Kaczmarz, S.: Angenäherte auflösung von Systemen linearer Gleichungen. *Bull. Internat. Acad. Polon. Sci. Lettres A*, 355–357 (1937)
20. Khan, U.A., Moura, J.M.: Distributed Kalman filters in sensor networks: bipartite fusion graphs. In: 2007 IEEE/SP 14th Workshop on Statistical Signal Processing, pp. 700–704. IEEE (2007)
21. Lan, G., Nemirovski, A., Shapiro, A.: Validation analysis of mirror descent stochastic approximation method. *Math. Program.* **134**(2), 425–458 (2012)
22. Loizou, N., Vaswani, S., Laradji, I.H., Lacoste-Julien, S.: Stochastic polyak step-size for sgd: an adaptive learning rate for fast convergence. In: International Conference on Artificial Intelligence and Statistics, pp. 1306–1314. PMLR (2021)
23. Lorenz, D.A., Schöpfer, F., Wenger, S.: The linearized Bregman method via split feasibility problems: analysis and generalizations. *SIAM. J. Imaging Sci.* **7**(2), 1237–1262 (2014)
24. Lorenz, D.A., Wenger, S., Schöpfer, F., Magnor, M.: A sparse Kaczmarz solver and a linearized bregman method for online compressed sensing. In: 2014 IEEE International Conference on Image Processing (ICIP), pp. 1347–1351. IEEE (2014)
25. Mertikopoulos, P., Lecouat, B., Zenati, H., Foo, C.S., Chandrasekhar, V., Piliouras, G.: Optimistic mirror descent in saddle-point problems: going the extra(-gradient) mile. In: International Conference on Learning Representations (2019)
26. Mertikopoulos, P., Staudigl, M.: Stochastic mirror descent dynamics and their convergence in monotone variational inequalities. *J. Optim. Theory Appl.* **179**(3), 838–867 (2018)
27. Miao, C.Q., Wu, W.T.: On greedy randomized average block Kaczmarz method for solving large linear systems. *J. Comput. Appl. Math.* **413**, 114,372 (2022)
28. Moorman, J.D., Tu, T.K., Molitor, D., Needell, D.: Randomized Kaczmarz with averaging. *BIT Numer. Math.* **61**(1), 337–359 (2021)
29. Necoara, I.: Faster randomized block Kaczmarz algorithms. *SIAM J. Matrix Anal. Appl.* **40**(4), 1425–1452 (2019)
30. Needell, D.: Randomized Kaczmarz solver for noisy linear systems. *BIT Numer. Math.* **50**(2), 395–403 (2010)
31. Needell, D., Tropp, J.A.: Paved with good intentions: analysis of a randomized block Kaczmarz method. *Linear Algebra Appl.* **441**, 199–221 (2014)
32. Nemirovski, A., Juditsky, A., Lan, G., Shapiro, A.: Robust stochastic approximation approach to stochastic programming. *SIAM J. Optim.* **19**(4), 1574–1609 (2009)
33. Nemirovski, A.S., Juditsky, A.B., Lan, G., Shapiro, A.: Robust stochastic approximation approach to stochastic programming. *SIAM J. Optim.* **19**(4), 1574–1609 (2009). <https://doi.org/10.1137/070704277>
34. Nemirovskij, A.S., Yudin, D.B.: *Problem Complexity and Method Efficiency in Optimization*. Wiley-Interscience (1983)
35. Nesterov, Y.: Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM J. Optim.* **22**(2), 341–362 (2012)
36. Olshanskii, M.A., Tyrtshnikov, E.E.: *Iterative Methods for Linear Systems: Theory and Applications*. SIAM (2014)
37. Patrascu, A., Necoara, I.: Nonasymptotic convergence of stochastic proximal point methods for constrained convex optimization. *J. Mach. Learn. Res.* **18**(1), 7204–7245 (2017)
38. Petra, S.: Randomized sparse block Kaczmarz as randomized dual block-coordinate descent. *Analele Stiintifice Ale Universitatii Ovidius Constanta-Seria Matematica* **23**(3), 129–149 (2015)
39. Popa, C.: Convergence rates for Kaczmarz-type algorithms. *Numer. Algorithms* **79**(1), 1–17 (2018). <https://doi.org/10.1007/s11075-017-0425-7>

40. Rabelo, J.C., Saporito, Y.F., Leitão, A.: On stochastic Kaczmarz type methods for solving large scale systems of ill-posed equations. *Inverse Probl.* **38**(2), 025003 (2022). <https://doi.org/10.1088/1361-6420/ac3f80>
41. Richtárik, P., Takáč, M.: Parallel coordinate descent methods for big data optimization. *Math. Program.* **156**(1), 433–484 (2016)
42. Richtárik, P., Takáč, M.: Stochastic reformulations of linear systems: algorithms and convergence theory. *SIAM J. Matrix Anal. Appl.* **41**(2), 487–524 (2020)
43. Schöpfer, F., Lorenz, D.A.: Linear convergence of the randomized sparse Kaczmarz method. *Math. Program.* **173**(1), 509–536 (2019). <https://doi.org/10.1007/s10107-017-1229-1>
44. Schöpfer, F., Lorenz, D.A., Tondji, L., Winkler, M.: Extended randomized Kaczmarz method for sparse least squares and impulsive noise problems. *Lineare Algebra Appl.* **652**, 132–154 (2022)
45. Stiefel, E.: Methods of conjugate gradients for solving linear systems. *J. Res. Nat. Bur. Standards* **49**, 409–435 (1952)
46. Strohmer, T., Vershynin, R.: A randomized Kaczmarz algorithm with exponential convergence. *J. Fourier Anal. Appl.* **15**(2), 262–278 (2009)
47. Tropp, J.A.: Improved analysis of the subsampled randomized Hadamard transform. *Advances Adapt. Data Anal.* **3**(01n02), 115–126 (2011)
48. Yin, W.: Analysis and generalizations of the linearized Bregman method. *SIAM J. Imaging Sci.* **3**(4), 856–877 (2010)
49. Zouzias, A., Freris, N.M.: Randomized extended Kaczmarz for solving least squares. *SIAM J. Matrix Anal. Appl.* **34**(2), 773–793 (2013)

**Publisher's note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

## Affiliations

Lionel Tondji<sup>1</sup> · Dirk A. Lorenz<sup>1</sup> 

Lionel Tondji  
l.ngoupeyou-tondji@tu-braunschweig.de

<sup>1</sup> Institute for Analysis and Algebra, TU Braunschweig, 38092 Braunschweig, Germany