Numer. Math. Theor. Meth. Appl. Vol. **12**, No. 1, pp. 1-20
doi: 10.4208/nmtma.OA-2018-0066 February 2019

Stochastic Gradient Descent for Linear Systems with Missing Data

Anna Ma[1,*] and Deanna Needell [2]

[1] *Claremont Graduate University, Claremont, CA 91711, USA*
[2] *University of California, Los Angeles, Los Angeles CA 90095, USA*

Received 25 May 2018; Accepted (in revised version) 21 July 2018

Abstract. Traditional methods for solving linear systems have quickly become impractical due to an increase in the size of available data. Utilizing massive amounts of data is further complicated when the data is incomplete or has missing entries. In this work, we address the obstacles presented when working with large data and incomplete data simultaneously. In particular, we propose to adapt the Stochastic Gradient Descent method to address missing data in linear systems. Our proposed algorithm, the Stochastic Gradient Descent for Missing Data method (mSGD), is introduced and theoretical convergence guarantees are provided. In addition, we include numerical experiments on simulated and real world data that demonstrate the usefulness of our method.

AMS subject classifications: 65F10, 65F20, 68W20

Key words: Linear systems, missing data, iterative methods, least squares problems.

1. Introduction

When handling large amounts of data, it may not be possible to load the entire matrix (data set) into memory, as typically required by matrix inversions or matrix factorization. This has led to the study and advancement of stochastic iterative methods with low memory footprints such as Stochastic Gradient Descent, Randomized Kaczmarz, and Randomized Gauss-Seidel [13, 16, 18, 23]. The need for algorithms that can process large amounts of information is further complicated by incomplete or missing data, which can arise due to, for example, attrition, errors in data recording, or cost of data acquisition. Standard methods for treating missing data, which include data imputation [6, 7], matrix completion [3, 11, 12, 19], and maximum likelihood estimation [5, 15] can be wasteful, create biases, or be impractical for extremely large amounts of data. This work simultaneously addresses both issues of large-scale and missing data.

[*] Corresponding author. *Email addresses:* a4ma@ucsd.edu (A. Ma), deanna@math.ucla.edu (D. Needell)

Consider the system of linear equations $\boldsymbol{Ax} = \boldsymbol{b}$[1], where $\boldsymbol{A} \in \mathbb{C}^{m\times n}$ is a large, full-rank, overdetermined ($m > n$) matrix. Suppose that $\boldsymbol{A}$ is not known entirely, but instead only some of its entries are available. As a concrete example, suppose $\boldsymbol{A}$ is the rating matrix from the survey of m users about n service questions, and $\boldsymbol{b}$ contains the m "overall" ratings from each user (which is fully known). Each user may not answer all of the individual service questions, but a company wishes to understand how each question affects the overall rating of the user. That is, given partial knowledge of $\boldsymbol{A}$, one wishes to uncover $\boldsymbol{x}_\star = \arg\min_{\boldsymbol{x}} \frac{1}{2m}\|\boldsymbol{Ax} - \boldsymbol{b}\|^2$.

Let $\tilde{\boldsymbol{A}} = \boldsymbol{D} \circ \boldsymbol{A}$ where $\boldsymbol{A}$ denotes the full matrix, and $\circ$ be the element-wise product, $\boldsymbol{D}$ denotes a binary matrix (1 indicating the availability of an element and 0 indicating a missing entry). Formally, one wants to solve the following optimization program:

$$\begin{aligned} &\text{Given } \ \tilde{\boldsymbol{A}}, \boldsymbol{b} \qquad \text{s.t. } \boldsymbol{Ax} = \boldsymbol{b} \ \textit{and} \ \tilde{\boldsymbol{A}} = \boldsymbol{D} \circ \boldsymbol{A}, \\ &\text{Find } \ \boldsymbol{x}_\star = \arg\min_{\boldsymbol{x}\in\mathscr{W}} \frac{1}{2m}\|\boldsymbol{Ax} - \boldsymbol{b}\|^2, \end{aligned} \tag{1.1}$$

where $\mathscr{W}$ is a convex domain containing the solution $\boldsymbol{x}_\star$ (e.g. a ball with large enough radius).

Contributions. This work presents a stochastic iterative projection method for solving large-scale linear systems with missing data. We provide theoretical bounds for the proposed method's performance and demonstrate its usefulness on simulated and real world data sets.

1.1. Stochastic Gradient Descent

Stochastic iterative methods such as Randomized Kaczmarz (RK) and Stochastic Gradient Descent (SGD) have gained interest in recent years due to their simplicity and ability to handle large-scale systems. Originally discussed in [20], SGD has proved to be particularly popular in machine learning [1, 2, 24]. SGD minimizes an objective function $F(\boldsymbol{x})$ over a convex domain $\mathscr{W}$ using unbiased estimates for the gradient of the objective, i.e., using $f_i(\boldsymbol{x})$ such that $\mathbb{E}[\nabla f_i(\boldsymbol{x})] = \nabla F(\boldsymbol{x})$. At each iteration, a random unbiased estimate, $\nabla f_i(\boldsymbol{x})$, is drawn and the minimizer of $F(\boldsymbol{x})$ is estimated with:

$$\boldsymbol{x}_k = \mathscr{P}_{\mathscr{W}}\left(\boldsymbol{x}_{k-1} - \alpha_k \nabla f_i(\boldsymbol{x}_{k-1})\right), \tag{1.2}$$

where α_k is an appropriately chosen step size, or learning rate, at iteration k and $\mathscr{P}_{\mathscr{W}}$ denotes the projection onto the convex set $\mathscr{W}$. To solve an overdetermined linear system $\boldsymbol{Ax} = \boldsymbol{b}$, one approach is to minimize the least-squares objective function $F(\boldsymbol{x}) = \frac{1}{2m}\|\boldsymbol{Ax} - \boldsymbol{b}\|^2 = \frac{1}{m}\sum_{i=1}^m f_i(\boldsymbol{x})$, where $f_i(\boldsymbol{x}) = \frac{1}{2}(\boldsymbol{A}_i\boldsymbol{x} - \boldsymbol{b}_i)^2$, $\boldsymbol{A}_i$ denotes the i^{th} row of $\boldsymbol{A}$, and $\boldsymbol{b}_i$ denotes the i^{th} entry of $\boldsymbol{b}$. In this setting, a random *row* of the matrix $\boldsymbol{A}$ is selected and

[1] The linear system is not assumed to be consistent; we will use the notation $\boldsymbol{Ax} = \boldsymbol{b}$ to denote a general linear system.

DOI https://doi.org/10.4208/nmtma.OA-2018-0066 | Generated on 2024-12-19 04:38:56

(1.2) is computed with $\nabla f_i(\boldsymbol{x}_{k-1}) = \boldsymbol{A}_i^*(\boldsymbol{A}_i\boldsymbol{x}_{k-1} - \boldsymbol{b}_i)$, where $(\cdot)^*$ denotes the conjugate transpose.

The performance of SGD on linear systems depends on the choice of α_k and the consistency of the system (i.e. whether a solution to the system exists). When the linear system is consistent, SGD achieves linear convergence with an appropriately chosen fixed step size [21]. For example, RK, a special instance of SGD for linear systems, has been shown to converge linearly for consistent systems without decreasing step sizes [10,18,23]. Unfortunately, this is not the case when the system is inconsistent. When the linear system is inconsistent, or $\boldsymbol{Ax} \approx \boldsymbol{b}$, one must use decreasing step sizes to obtain the optimum (see e.g. [4,9,21]). This phenomenon is explained by the norm of the unbiased estimates at the minimizer, $\|\nabla f_i(\boldsymbol{x}_\star)\|^2$. For consistent systems, $\|\nabla f_i(\boldsymbol{x}_\star)\|^2 = \|\boldsymbol{A}_i^*(\boldsymbol{A}_i\boldsymbol{x}_\star - \boldsymbol{b}_i)\|^2 = 0$ since $\boldsymbol{Ax}_\star = \boldsymbol{b}$. Intuitively, as SGD progresses closer to the minimizer, the magnitude of the iterates get smaller and allow SGD to converge. When the system is inconsistent, $\boldsymbol{Ax}_\star = \boldsymbol{b} + \boldsymbol{r}$ for some residual vector $\boldsymbol{r}$ and least squares minimizer $\boldsymbol{x}_\star$. As the SGD approximates approach $\boldsymbol{x}_\star$, the magnitude of the iterates do not converge to 0 since $\|\nabla f_i(\boldsymbol{x}_\star)\|^2 = \|\boldsymbol{A}_i^*(\boldsymbol{A}_i\boldsymbol{x}_\star - \boldsymbol{b}_i)\|^2 = \boldsymbol{r}_i^2\|\boldsymbol{A}_i\|^2$. Using diminishing step sizes dampens the magnitude of the iterates over time, allowing SGD to converge. When SGD with fixed step size is applied to inconsistent systems, the iterates oscillate within a fixed distance from the solution [18]. The fixed distance, also referred to as the convergence horizon, is proportional to the step size but inversely proportional to the rate of convergence. Therefore, there is a trade-off between the rate of convergence (speed) and the radius of convergence (accuracy).

The proposed method, which we refer to as mSGD, is an SGD-type iterate with a correction term that takes into account the fact that not all entries of $\boldsymbol{A}$ are available. We start with a discussion on the model under which mSGD operates and proceed to derive the iterate. After the introduction of the algorithm, the formal results are stated.

Outline. Section 2 introduces the proposed method, the Stochastic Gradient Descent for Missing Data method (mSGD) and the main theoretical results. The performance of mSGD on simulated and real world data are shown in Section 3. Finally, we conclude in Section 4.

2. Stochastic Gradient Descent for missing data

We model whether an entry of $\boldsymbol{A}$ is missing with i.i.d. Bernoulli random variables that are equal to 1 with probability p. Practically, there are many applications in which this type of assumption holds. For example, surveys where participants are given a random subset of questions to answer follow this assumption. In collaborative filtering, there are various models where such assumptions hold [8,17]. As another example, consider an extremely large $m \times n$ matrix $\boldsymbol{A}$ where it is not possible to load entire rows of $\boldsymbol{A}$ nor columns of $\boldsymbol{A}$ due to memory constraints. Instead, one is restricted to only loading $\lfloor pn \rfloor$ (random) elements of $\boldsymbol{A}$ at a time. Under this probabilistic assumption on the missing entries, the least squares solution can be computed without making any additional assumptions on the structure of $\boldsymbol{A}$ such as sparsity or low-rankness. In the case of having a fixed matrix $\tilde{\boldsymbol{A}} \in \mathbb{C}^{m\times n}$ with missing entries, the theoretical results hold only if each row of the matrix is utilized once. If $\tilde{\boldsymbol{A}}$ is an extremely overdetermined matrix (i.e. $m \gg n$), then this is a reasonable assumption.

Notation. Let $\boldsymbol{D}$ be an $m \times n$ matrix where the entries of $\boldsymbol{D}$, denoted by $\delta_{i,j}$ for $i = 1, 2, \cdots, m$ and $j = 1, 2, \cdots, n$, are drawn independent and identically distributed (i.i.d.) from a Bernoulli distribution with parameter p so that $\delta_{i,j} = 1$ with probability p. The matrix $\boldsymbol{D}$ is referred to as a *binary mask* throughout and its entries indicate the locations of non-missing entries of $\boldsymbol{A}$. Let $\boldsymbol{D}_i$ be the diagonal matrix whose diagonal is equal to the ith row of $\boldsymbol{D}$. Given an $n \times n$ matrix $\boldsymbol{M}$, we denote the a matrix containing only the diagonal of $\boldsymbol{M}$ as $\mathrm{diag}(\boldsymbol{M})$. Let $\tilde{\boldsymbol{A}}$ represent the matrix $\boldsymbol{A}$ with missing elements filled in with zeros so that $\tilde{\boldsymbol{A}} = \boldsymbol{D} \circ \boldsymbol{A}$ and $\tilde{\boldsymbol{A}}_i = \boldsymbol{D}_i \boldsymbol{A}_i^*$, where $\circ$ denotes the element-wise product. Additionally, let $\sigma^2_{\min}(\boldsymbol{A})$ be the smallest singular value of $\boldsymbol{A}$ and $\|\cdot\|$ denote the ℓ_2-norm. The expected value taken over the random selection of rows of $\tilde{\boldsymbol{A}}$ is denoted $\mathbb{E}_i[\cdot]$, the expected value taken over all (2^{mn}) possible binary masks $\boldsymbol{D}$ as $\mathbb{E}_\delta[\cdot]$, and the full expected value as $\mathbb{E}[\cdot]$. Lastly, let $\mathscr{W}$ be some convex domain containing $\boldsymbol{x}_\star$ and $B := \max_{\boldsymbol{x}\in\mathscr{W}} \|\boldsymbol{x}\|^2$.

2.1. The method

Suppose one naively applies SGD to the system $\tilde{\boldsymbol{A}}\boldsymbol{x} = \boldsymbol{b}$. To that end, consider the objective $\widehat{F} = \frac{1}{2m}\|\tilde{\boldsymbol{A}}\boldsymbol{x} - \boldsymbol{b}\|^2 = \frac{1}{m}\sum_{i=1}^m \widehat{f}_i(\boldsymbol{x})$ where $\widehat{f}_i(\boldsymbol{x}) = \frac{1}{2}(\tilde{\boldsymbol{A}}_i\boldsymbol{x} - \boldsymbol{b}_i)^2$. This objective function leads to the update:

$$\boldsymbol{x}_k = \boldsymbol{x}_{k-1} - \alpha_k \left(\tilde{\boldsymbol{A}}_i^*(\tilde{\boldsymbol{A}}_i\boldsymbol{x}_{k-1} - \boldsymbol{b}_i)\right),$$

since $\nabla\widehat{f}_i(\boldsymbol{x}) = \tilde{\boldsymbol{A}}_i^*(\tilde{\boldsymbol{A}}_i\boldsymbol{x} - \boldsymbol{b}_i)$. Unfortunately, one computes that, taking the expectation with respect to the binary mask and gradient direction,

$$\begin{aligned}&\mathbb{E}_i\mathbb{E}_\delta[\nabla\widehat{f}_i(\boldsymbol{x})] = \mathbb{E}_i\mathbb{E}_\delta[\tilde{\boldsymbol{A}}_i^*(\tilde{\boldsymbol{A}}_i\boldsymbol{x} - \boldsymbol{b}_i)]\\ =&\frac{1}{m}\left(p^2\boldsymbol{A}^*\boldsymbol{A}\boldsymbol{x} + (p-p^2)\mathrm{diag}(\boldsymbol{A}^*\boldsymbol{A})\boldsymbol{x} - p\sum_i \boldsymbol{A}_i^*\boldsymbol{b}_i\right) \neq \nabla F(\boldsymbol{x}).\end{aligned}$$

As a result, the iterates are not moving in the gradient descent direction toward the desired solution in expectation.

Now, since we have information on the distribution of missing entries, we can use this to design a better objective function. For example, we can approximate the proportion of the right hand side vector $\boldsymbol{b}$ which can be accounted for using the distribution for missing entries. In other words, since $\mathbb{E}_\delta[\tilde{\boldsymbol{A}}_i\boldsymbol{x}] = p\boldsymbol{b}_i$, consider the objective $\tilde{F}(\boldsymbol{x}) = \|\tilde{\boldsymbol{A}}\boldsymbol{x} - p\boldsymbol{b}\|_2^2$. Applying SGD to this objective, one computes

$$\begin{aligned}&\mathbb{E}_i\mathbb{E}_\delta[\nabla\tilde{f}_i(\boldsymbol{x})] = \mathbb{E}_i\mathbb{E}_\delta[\tilde{\boldsymbol{A}}_i^*(\tilde{\boldsymbol{A}}_i\boldsymbol{x} - p\boldsymbol{b}_i)]\\ =&\frac{1}{m}\left(p^2\boldsymbol{A}^*\boldsymbol{A}\boldsymbol{x} + (p-p^2)\mathrm{diag}(\boldsymbol{A}^*\boldsymbol{A})\boldsymbol{x} - p^2\sum_i \boldsymbol{A}_i^*\boldsymbol{b}_i\right) \neq \nabla F(\boldsymbol{x}),\end{aligned}$$

which is again not the direction that one wants on average.

Instead of using $\nabla\tilde{f}(\boldsymbol{x})$ as the step direction, we use $\nabla\tilde{f}_i(\boldsymbol{x})$ to estimate $\nabla F(\boldsymbol{x})$. In other words, we want to represent $\nabla F(\boldsymbol{x})$ in terms of $\mathbb{E}[\nabla\tilde{f}_i(\boldsymbol{x})]$. By doing so, iterates $\boldsymbol{x}_k$ move in the gradient descent direction towards the least squares solution to the objective $F(\boldsymbol{x}) = \frac{1}{2m}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|^2$. From the above computation, one can see

$$\nabla F(\boldsymbol{x}) = \frac{1}{p^2}\mathbb{E}[\nabla\tilde{f}_i(\boldsymbol{x})] - \frac{(1-p)}{p^2}\mathbb{E}[\mathrm{diag}(\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i)]\boldsymbol{x}.$$

The detailed computation is available in the Appendix (Lemma A.1). Therefore the appropriate update is

$$\boldsymbol{x}_k = \boldsymbol{x}_{k-1} - \alpha_k\left(\frac{1}{p^2}\left(\tilde{\boldsymbol{A}}_i^*(\tilde{\boldsymbol{A}}_i\boldsymbol{x}_{k-1} - p\boldsymbol{b}_i)\right) - \frac{1-p}{p^2}\mathrm{diag}(\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i)\boldsymbol{x}_{k-1}\right).$$

Note that in classical SGD literature, the expected value is taken over the row choice i when being applied to linear systems. However, in this setting there are two sources of randomness: the randomness from row selection and the randomness incurred by modeling missing data. In this computation, the expected value is taken with respect to both sources of randomness. The method is outlined in Algorithm 2.1.

Algorithm 2.1 Stochastic Gradient Descent for Missing Data (mSGD).

```
1: procedure ($\tilde{A}$, $b$, $T$, $p$, $\{\alpha_k\}$)        ▷ If using a fixed step size $\alpha$, $\alpha_k = \alpha$ for all $k$.
2:     Initialize $x_0$
3:     for $k = 1, 2, \cdots, T$ do
4:         Choose row $i$ of $\tilde{A}$ with probability $\frac{1}{m}$
5:         $g(x_{k-1}) = \frac{1}{p^2}\left(\tilde{A}_i^*(\tilde{A}_i x_{k-1} - p b_i)\right) - \frac{1-p}{p^2}\mathrm{diag}(\tilde{A}_i^*\tilde{A}_i)x_{k-1}$
6:         $x_k = \mathcal{P}_{\mathcal{W}}\left(x_{k-1} - \alpha g(x_{k-1})\right)$        ▷ $\mathcal{P}_{\mathcal{W}}$ is the projection onto the set $\mathcal{W}$.
7:     end for
8:     Output $x_k$
9: end procedure
```

2.2. Main results

Before the main results are presented, note the following properties of the objective function,

$$F(\boldsymbol{x}) = \frac{1}{2m}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|^2, \tag{2.1}$$

and the update function in Algorithm 2.1 (Line 5),

$$g(\boldsymbol{x}) = \frac{1}{p^2}\left(\tilde{\boldsymbol{A}}_i^*(\tilde{\boldsymbol{A}}_i\boldsymbol{x} - p\boldsymbol{b}_i)\right) - \frac{(1-p)}{p^2}\mathrm{diag}(\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i)\boldsymbol{x}, \tag{2.2}$$

as they play an important role in the convergence analysis of mSGD.

- The objective function (2.1) is μ-strongly convex. For all $\boldsymbol{x},\boldsymbol{y}\in\mathscr{W}$,

  $$(\boldsymbol{x}-\boldsymbol{y})^*(\nabla F(\boldsymbol{x})-\nabla F(\boldsymbol{y}))\geq\mu\|\boldsymbol{x}-\boldsymbol{y}\|^2,$$

  where
  $$\mu=\frac{\sigma_{min}^2(\boldsymbol{A})}{m}. \tag{2.3}$$

- The update function $g(\boldsymbol{x})$ is Lipschitz continuous, has Lipschitz constant $L_{i,D}$ (for a fixed instance of i and D), and supremum Lipschitz constant L_g. In other words, for all $\boldsymbol{x},\boldsymbol{y}\in\mathscr{W}$,

  $$\|g(\boldsymbol{x})-g(\boldsymbol{y})\|\leq L_{i,D}\|\boldsymbol{x}-\boldsymbol{y}\|, \tag{2.4a}$$
  $$L_g=\sup_{i,D}L_{i,D}. \tag{2.4b}$$

  The supremum is taken over all choices of rows and all possible binary masks (i.e. all 2^{mn} possible binary masks).

- There exists a constant G that uniformly bounds the expected norm of $\|g(\boldsymbol{x})\|^2$,

  $$\mathbb{E}[\|g(\boldsymbol{x})\|^2]\leq G \tag{2.5}$$

  for all $\boldsymbol{x}\in\mathscr{W}$ and rows $\tilde{\boldsymbol{A}}_i$. The expected norm of $g(\boldsymbol{x}_\star)$ plays an important role in the convergence horizon. For this reason, let $G_\star$ denote the upper bound of $\mathbb{E}[\|g(\boldsymbol{x}_\star)\|^2]$:

  $$\mathbb{E}[\|g(\boldsymbol{x}_\star)\|^2]\leq G_\star. \tag{2.6}$$

The computation of $L_{i,D}$ and L_g are shown in Lemma A.2. Lemma A.3 shows the computation of G and $G_\star$. The statements and proofs of both lemmas are provided in the Appendix so that we may proceed to the presentation of the main results.

Theorem 2.1 shows that, in expectation, Algorithm 2.1 converges to the least squares solution of the linear system $\boldsymbol{Ax}=\boldsymbol{b}$ with properly chosen step size. This theorem is an application of the previously proven result stated in Lemma 2.1. The fixed step size regime and the trade off between convergence rate and accuracy is explored in Theorem 2.2. In addition, we provide an optimal step size choice based on a desired error tolerance, ϵ, and a bound on the number of iterations required to obtain said tolerance in Corollary 2.1. Lastly, we remark on the recovery of classical SGD when $p=1$ both algorithmically and with respect to the proven error bounds.

Theorem 2.1. *Consider* (1.1) *with* $\tilde{\boldsymbol{A}}=\boldsymbol{D}\circ\boldsymbol{A}$ *where entries of* $\boldsymbol{D}$ *are drawn i.i.d. from a Bernoulli distribution with probability parameter* p*. Let* μ *be as defined in* (2.3)*. Choosing* $\alpha_k=\frac{1}{\mu k}$*, Algorithm* 2.1 *converges in expectation with error*

$$\mathbb{E}[\|\boldsymbol{x}_k-\boldsymbol{x}_\star\|^2]\leq\frac{17G(1+\log(k))}{\mu^2k},$$

DOI https://doi.org/10.4208/nmtma.OA-2018-0066 | Generated on 2024-12-19 04:38:56

where

$$G = \frac{2B}{mp^2}\left(1 + \frac{(1-p)(2-p)}{p}\right)\sum_i \left\|\boldsymbol{A}_i\right\|^4 + \frac{2}{mp^2}\sum_i \left\|\boldsymbol{A}_i\right\|^2 |\boldsymbol{b}_i|^2$$

is an upper bound on $\mathbb{E}[\|g(x)\|^2]$ *and* $B = \max_{\boldsymbol{x}\in\mathscr{W}} \|\boldsymbol{x}\|^2$.

It is clear that the convergence behavior of Algorithm 2.1 depends on G, the uniform upper bound on the expected norm of $g(\boldsymbol{x})$ and on $\sigma^2_{min}(\boldsymbol{A})$. As one would expect, the more data that is missing, the larger the upper bound on expected error. In particular, assuming all other variables are constant and $p \in (0,1]$, as p decreases, G increases. Theorem 2.1 is an application of the following previously proved lemma.

Lemma 2.1. ([22, Theorem 1]) *Let* $F(\boldsymbol{x})$ *be a* μ*-strongly convex objective function,* $g(\boldsymbol{x})$ *be such that* $\mathbb{E}[g(\boldsymbol{x})] = \nabla F(\boldsymbol{x})$*, and* $\mathbb{E}[\|g(\boldsymbol{x})\|^2] \le G$ *for all* $\boldsymbol{x} \in \mathscr{W}$*. Using step size* $\alpha_k = \frac{1}{\mu k}$ *and update* $\boldsymbol{x}_k = \mathscr{P}_{\mathscr{W}}(\boldsymbol{x}_{k-1} - \alpha_k g(\boldsymbol{x}_{k-1}))$*, it holds that*

$$\mathbb{E}[F(\boldsymbol{x}_k) - F(\boldsymbol{x}_\star)] \le \frac{17G(1+\log(k))}{\mu k}.$$

The next theorem details the convergence behavior of Algorithm 2.1 when using a fixed step size. Theorem 2.2 shows that Algorithm 2.1 experiences a convergence horizon that depends on L_g and $G_\star$. For $p \in (0,1]$, as p decreases, $G_\star$ and L_g both increase. Intuitively this makes sense as a larger amount of missing data should increase the size of the convergence horizon. Additionally, the convergence rate $r = \left(1 - 2\alpha\mu\left(1 - \alpha L_g\right)\right)$ also increases as p decreases. In other words, more missing data causes a slower convergence rate.

Theorem 2.2. *Consider* (1.1) *with* $\tilde{\boldsymbol{A}} = \boldsymbol{D} \circ \boldsymbol{A}$ *where entries of* $\boldsymbol{D}$ *are drawn i.i.d. from a Bernoulli distribution and are equal to 1 with probability* p*. Let* L_g*,* $G_\star$*, and* μ *be as defined in* (2.4b)*,* (2.6)*, and* (2.3) *respectively. Additionally, let the fixed step size be* $\alpha < \frac{1}{L_g}$*. Algorithm* 2.1 *converges with expected error*

$$\mathbb{E}\left[\|\boldsymbol{x}_k - \boldsymbol{x}_\star\|^2\right] \le r^k\|\boldsymbol{x}_0 - \boldsymbol{x}_\star\|^2 + \frac{\alpha G_\star}{\mu\left(1 - \alpha L_g\right)}, \tag{2.7}$$

where $r = \left(1 - 2\alpha\mu\left(1 - \alpha L_g\right)\right)$*,* $L_g = \frac{1}{p^2}\sup_i \|\boldsymbol{A}_i\|^2$*,* $\mu = \sigma^2_{min}(\boldsymbol{A})/m$*. If* $\boldsymbol{Ax} = \boldsymbol{b}$ *is consistent,*

$$G_\star = \frac{2(1-p)(2-p)}{mp^3}\|\boldsymbol{x}_\star\|^2\sum_i \left\|\boldsymbol{A}_i\right\|^4.$$

If the linear system is inconsistent (i.e. $\boldsymbol{Ax} = \boldsymbol{b} + \boldsymbol{r}$ *for some residual vector* $\boldsymbol{r}$*), then*

$$G_\star = \frac{2}{mp^2}\sum_i \left\|\boldsymbol{A}_i\right\|^2 \boldsymbol{r}_i^2 + \frac{2(1-p)(2-p)}{mp^3}\|\boldsymbol{x}_\star\|^2\sum_i \left\|\boldsymbol{A}_i\right\|^4.$$

Corollary 2.1 and the subsequent remark comment on the number of iterations required by Algorithm 2.1 to obtain some desired error tolerance ϵ using a particular fixed step size α^*. The corollary itself details this information in terms of the variables in Theorem 2.2 while the remark translates and simplifies α^* and k (the number of iterations) into terms relating to $\boldsymbol{A}$. Note that the number of iterations required to reach a specified tolerance is a function of the ratio between the log of the initial error and ϵ. The number of iterations increase as ϵ decreases. Additionally, the remark shows that as p decreases, or as less data becomes available, more iterations are required to obtain an expected error of ϵ. The proof of Corollary 2.1 can be found in ([18] Corollary 2.2) with different constants.

Corollary 2.1. *Given an initial error ϵ_0 and choosing the fixed step size*

$$\alpha^* = \frac{\epsilon\mu}{2G_\star + 2\mu\epsilon L_g},$$

after

$$k = 2\log\left(\frac{2\epsilon_0}{\epsilon}\right)\left(\frac{L_g}{\mu} + \frac{G_\star}{\mu^2\epsilon}\right)$$

iterations of Algorithm 2.1*,* $\mathbb{E}[\|\boldsymbol{x}_k - \boldsymbol{x}_\star\|^2] \le \epsilon$ *holds in expectation.*

Remark 2.1. Let $\boldsymbol{a}^2_{\max} = \max_i \|\boldsymbol{A}_i\|^2$ be the maximum squared row norm of $\boldsymbol{A}$. Given an initial error ϵ_0 and a desired tolerance ϵ to the true solution, choosing the fixed step size

$$\alpha^* = \frac{p^3\epsilon\sigma^2_{\min}(\boldsymbol{A})}{4(2-p)(1-p)\|\boldsymbol{x}_\star\|^2\sum_i\|\boldsymbol{A}_i\|^4 + 2p\epsilon\boldsymbol{a}^2_{\max}\sigma^2_{\min}(\boldsymbol{A})},$$

after

$$k = 2\log\left(\frac{2\epsilon_0}{\epsilon}\right)\left(\frac{m\boldsymbol{a}^2_{\max}}{p^2\sigma^2_{\min}(\boldsymbol{A})} + \frac{2(2-p)(1-p)m\|\boldsymbol{x}_\star\|^2\sum_i\|\boldsymbol{A}_i\|^4}{p^3\sigma^4_{\min}(\boldsymbol{A})\epsilon}\right)$$

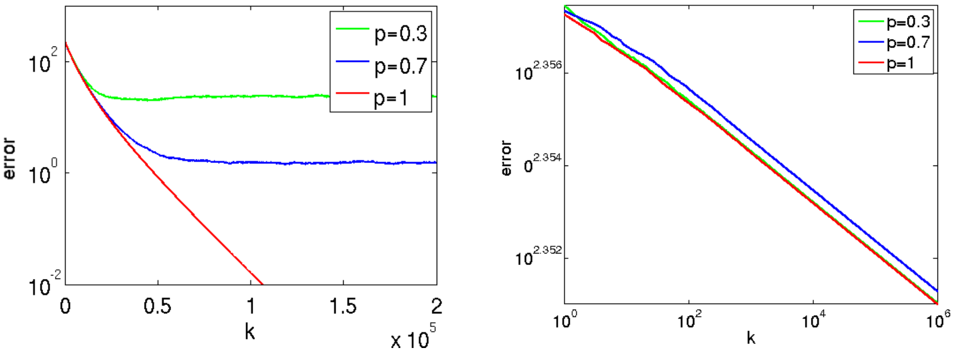
iterations of Algorithm 2.1, $\mathbb{E}[\|\boldsymbol{x}_k - \boldsymbol{x}_\star\|^2] \le \epsilon$ holds in expectation.

Recovering SGD. When $p = 1$, Algorithm 2.1 behaves as classical SGD does on the full linear system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$. Additionally, mSGD experiences similar convergence bounds as classical SGD for fixed step sizes [18]. In particular, when $p = 1$ the updating function $g(\boldsymbol{x})$ reduces to $g(\boldsymbol{x}) = \boldsymbol{A}^*_i(\boldsymbol{A}_i\boldsymbol{x} - \boldsymbol{b}_i)$.

3. Experiments

This section demonstrates the usefulness of Algorithm 2.1 on synthetic and real world data. Although the full data set is available in every experiment, missing data is simulated by computing a binary mask that dictates which elements are available at every iteration. By doing so, the simplifying assumption is satisfied, and the ground truth is known, approximation error is computable, and we can investigate the performance of Algorithm 2.1 with varying levels of missing data. In each experiment, the percentage of available data is

DOI https://doi.org/10.4208/nmtma.OA-2018-0066 | Generated on 2024-12-19 04:38:56



(a) (b)

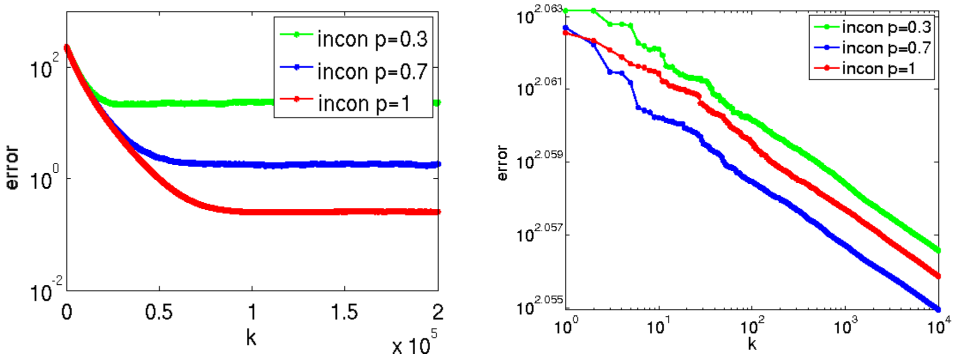
Figure 1: This figure compares the performance of Algorithm 2.1 on linear systems drawn from a standard Gaussian distribution. The percentage of data that is missing is varied. The $x-$axis is log(iteration) and the $y-$axis is the log(ℓ_2-error). Note that using a fixed step size (left), allows mSGD to converge much faster but to some convergence horizon. Using updating step sizes (right), continual progress is made at the cost of slower convergence.

varied and log ℓ_2-error to the least squares solution, $||\boldsymbol{x}_k - \boldsymbol{x}_\star||^2$ is averaged over 20 trials. For the fixed step size in simulated data, $\alpha = 10^{-4}$ and for real world data $\alpha = 10^{-5}$. For the updating step size regime, $\alpha_k = \frac{c}{\sigma^2_{\min}(A)k}$ with $c = 10^{-2}$. Using $\alpha_k = \frac{1}{\sigma^2_{\min}(\boldsymbol{A})k}$ (as described in Theorem 2.1) creates an initial increase in error followed by a decrease in error. This behavior is attributed to the step sizes being too large initially. It seems that the factor c can be optimized but we do not attempt to optimize such parameters here.

In the first experiment, we apply mSGD to synthetic data. The results can be seen in Fig. 1 and Fig. 2. Here, elements of $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ are drawn i.i.d. from a standard Gaussian distribution where $m = 1000$ and $n = 200$. Fig. 1(a) and Fig. 2(a) show the results of Algorithm 2.1 using a fixed step size ($\alpha = 10^{-4}$) while Fig. 1(b) and Fig. 2(b) show results using updating step sizes. For inconsistent systems, we use $\boldsymbol{b} + \boldsymbol{r}$ as the right hand side vector where $\boldsymbol{r}$ is computed such that $\boldsymbol{r} \in \text{null}(\boldsymbol{A}^*)$ using Matlab's `null()` function.

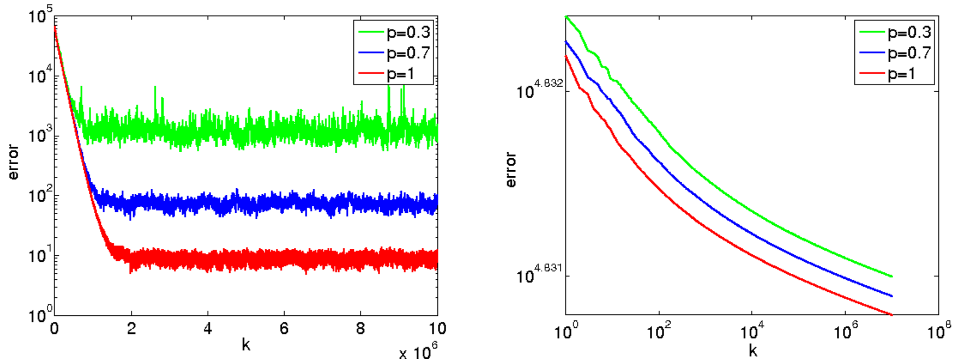
The first real world data set was obtained form the UCI Machine Learning Repository [14] and contains data from a bike rental service. Rows of $\boldsymbol{A}$ contain hourly information from a bike share rental system and columns contain information such as weather, total number of rented bikes, time, and day of the week. In this experiment, $m = 17379$ and $n = 9$. Fig. 3 displays the performance of mSGD on this data set for fixed and updating step sizes.

The performance of Algorithm 2.1 on Lyme data from lymedisease.org is shown in Fig. 4. This data set contains survey responses from patients who have been diagnosed with Lyme Disease. Examples of responses include number of emergency room visits, severity of symptoms, and effectiveness of medication. For the right hand side vector, we use the number of health care providers a patient saw before being diagnosed with Lyme. For this experiment, $m = 3686$ and $n = 81$. Solving such a system would potentially uncover what factors lead to late-stage diagnosis, a critical question in Lyme disease research. As seen



(a) (b)

Figure 2: The performance of mSGD on inconsistent linear systems. For a fixed step size (left), mSGD converges to a convergence horizon and using updating step sizes (right) allows mSGD to continually progress at a slower rate.



(a) (b)

Figure 3: For bike data set, mSGD with a fixed step size (left) experiences a congerence horizon. Using updating step sizes (right), mSGD continues to progress toward the least squares solution.

in Fig. 2(b), when using the updating step size $\alpha_k = \frac{c}{\sigma_{\min}^2(\boldsymbol{A})k}$, the convergence rate suffers because the step size decays too quickly. Theoretically, we expect the error to continue to decay very slowly but, practically, it makes more sense to use another updating step size regime. In this experiment, we instead use $\alpha_k = \frac{c}{\sigma_{\min}^2(\boldsymbol{A})} r^{\lfloor k/T* \rfloor}$ so that the initial step size is $\frac{c}{\sigma_{\min}^2(\boldsymbol{A})}$ and after every T^* iterations, the step size is multiplied by a factor of $r < 1$. Empirical parameter tuning led us to use $c = 10^{-3}$ for $p = 0.7$ and $p = 1$, $c = 10^{-4}$ for $p = 0.3$, $T^* = 10^5$, and $r = 0.8$. The results are shown in Fig. 4.

Fig. 5 compares mSGD and classical SGD applied to three different imputation treatments for missing data. We use the Lyme Disease data set with updating step sizes as described in the previous experiment with $c = 10^{-4}$ and $T^* = 10^5$. Setting $p = 0.5$, $\boldsymbol{A} \in \mathbb{R}^{10^5 \times 81}$ where each row of $\boldsymbol{A}$ is a randomly selected row of the Lyme Disease data set

DOI https://doi.org/10.4208/nmtma.OA-2018-0066 | Generated on 2024-12-19 04:38:56

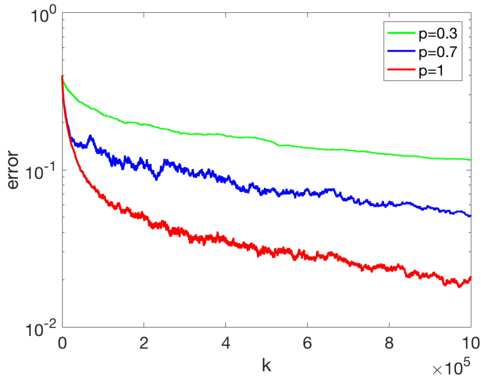


Figure 4: Using updating step sizes, Algorithm 2.1 has decaying approximation error to the least squares solution of the completed linear system.

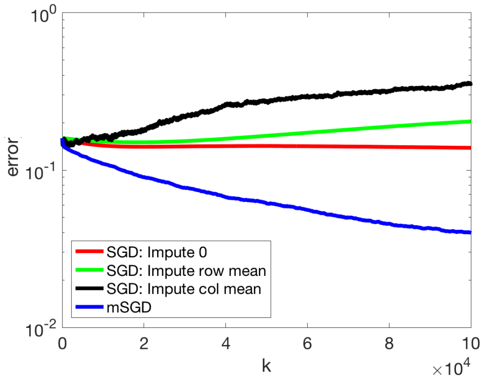


Figure 5: The proposed algorithm out performs using imputation methods with SGD.

with roughly half of the entries of the row (randomly) removed. Classical SGD is applied to $\tilde{\boldsymbol{A}}$ in three ways: imputing 0 (if $\tilde{\boldsymbol{A}}_{ij}$ is missing, $\tilde{\boldsymbol{A}}_{ij} = 0$), imputing row means (if $\tilde{\boldsymbol{A}}_{ij}$ is missing, $\tilde{\boldsymbol{A}}_{ij}$ is the average over all non-missing elements in $\tilde{\boldsymbol{A}}_i$), and imputing column means (if $\tilde{\boldsymbol{A}}_{ij}$ is missing, $\tilde{\boldsymbol{A}}_{ij}$ is the average over all non-missing elements in the j^{th} column of $\tilde{\boldsymbol{A}}$). Notice that mSGD outperforms the imputation methods presented here.

These experimental results support the theoretical findings presented in Section 2. Using a fixed step size, mSGD converges to some radius around the solution while using updating step size allows us to avoid the convergence horizon at the price of a slower convergence. For fixed step size, the amount of missing data affects the convergence horizon. In particular, as p decreases the size of the convergence horizon increases.

4. Conclusions

In this work, we present a stochastic iterative projection method that solves linear systems with missing data. We prove that mSGD finds the least squares solution to the

DOI https://doi.org/10.4208/nmtma.OA-2018-0066 | Generated on 2024-12-19 04:38:56

linear system with full data even though a system has missing data. Additionally, this work shows theoretical bounds the performance of mSGD using fixed and updating step sizes. The experiments show that the proposed method is useful in real world settings when one wishes to solve a linear system with missing data without needing to impute missing values, which can be extremely costly.

Appendix A

Consider the objective functions

$$F(\boldsymbol{x})=\frac{1}{2m}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|^2=\frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}(\boldsymbol{A}_i\boldsymbol{x}-\boldsymbol{b}_i)^2,\qquad \tilde{F}(\boldsymbol{x})=\frac{1}{2m}\|\tilde{\boldsymbol{A}}\boldsymbol{x}-p\boldsymbol{b}\|^2.$$

Let $\tilde{f}_i(\boldsymbol{x})=\frac{1}{2}(\tilde{\boldsymbol{A}}_i\boldsymbol{x}-p\boldsymbol{b}_i)^2$. Let $\mathbb{E}_\delta[\cdot]$ denote the expected value function with respect to the Bernoulli random variables of the binary mask $\boldsymbol{D}$ and $\mathbb{E}_i[\cdot]$ denote the expected value with respect to the choice of rows of $\tilde{\boldsymbol{A}}$. In addition, let μ be the strong convexity parameter $F(\boldsymbol{x})$ so that for any $\boldsymbol{x},\boldsymbol{y}\in\mathscr{W}$, $(\boldsymbol{x}-\boldsymbol{y})^*(\nabla F(\boldsymbol{x})-\nabla F(\boldsymbol{y}))\geq\|\boldsymbol{x}-\boldsymbol{y}\|^2\mu$.

First, we will show a few useful properties pertaining to the update function $g(\boldsymbol{x})$. In particular, Lemma A.1 shows that in expectation $g(\boldsymbol{x})$ allows us to make progress in the gradient direction of the objective $F(\boldsymbol{x})$ (as opposed to the direction of $\nabla\tilde{F}(\boldsymbol{x})$). Next, Lemma A.2 investigates the Lipschitz continuity of $g(\boldsymbol{x})$ for a fixed row i and binary mask $\boldsymbol{D}$ and its supremum Lipschitz constant of $g(\boldsymbol{x})$ over all rows and binary masks. Lemma A.3 shows that we can uniformly bound the expected norm of $g(\boldsymbol{x})$ and provides said bound. Finally, we prove Theorem 2.2.

Lemma A.1. *The expected value of the update function $g(\boldsymbol{x})$ defined in* (2.2) *is the gradient of the objection function $F(\boldsymbol{x})$. In other words, for*

$$g(\boldsymbol{x})=\frac{1}{p^2}\left(\tilde{\boldsymbol{A}}_i^*(\tilde{\boldsymbol{A}}_i\boldsymbol{x}-p\boldsymbol{b}_i)\right)-\frac{(1-p)}{p^2}\mathrm{diag}(\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i)\boldsymbol{x},$$

we have that $\mathbb{E}[g(\boldsymbol{x})]=\nabla F(\boldsymbol{x})$.

Proof. To prove this lemma, we will first take the expected value of $\nabla\tilde{f}_i(\boldsymbol{x})$. We then take the expected value of $g(\boldsymbol{x})$, substitute $\mathbb{E}[\nabla\tilde{f}_i(\boldsymbol{x})]$, and simplify to complete the proof. Let's first check that

$$\mathbb{E}[\nabla\tilde{f}_i(\boldsymbol{x})]=\mathbb{E}[\tilde{\boldsymbol{A}}_i^*(\tilde{\boldsymbol{A}}_i\boldsymbol{x}-p\boldsymbol{b}_i)]=p^2\boldsymbol{A}^*\boldsymbol{A}\boldsymbol{x}+(p-p^2)\mathrm{diag}(\boldsymbol{A}^*\boldsymbol{A})\boldsymbol{x}-p^2\sum_i\boldsymbol{A}_i\boldsymbol{b}_i. \tag{A.1}$$

Taking a simple derivative, $\nabla\tilde{f}_i(\boldsymbol{x})=\tilde{\boldsymbol{A}}_i^*(\tilde{\boldsymbol{A}}_i\boldsymbol{x}-p\boldsymbol{b}_i)$. The matrix $\boldsymbol{D}$ is a $m\times n$ binary mask with entries $\delta_{i,j}\overset{i.i.d.}{\sim}Bern(p)$. Let $\boldsymbol{D}_i=\mathrm{diag}(\delta_{i,1},\delta_{i,2},\cdots,\delta_{i,n})$ be a $n\times n$ diagonal matrix so that $\tilde{\boldsymbol{A}}_i=\boldsymbol{D}_i\boldsymbol{A}_i^*$. Substituting $\tilde{\boldsymbol{A}}_i=\boldsymbol{D}_i\boldsymbol{A}_i^*$ and taking the expectation with respect to the $\delta_{i,j}$'s,

$$\begin{aligned}&\mathbb{E}_\delta[\nabla\tilde{f}_i(\boldsymbol{x})]=\mathbb{E}_\delta[\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i]\boldsymbol{x}-p\mathbb{E}_\delta[\tilde{\boldsymbol{A}}_i^*]\boldsymbol{b}_i\\ =&\mathbb{E}_\delta[\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i]\boldsymbol{x}-p^2\boldsymbol{A}_i^*\boldsymbol{b}_i\overset{(i)}{=}p^2\boldsymbol{A}_i^*\boldsymbol{A}_i\boldsymbol{x}+(p-p^2)\mathrm{diag}(\boldsymbol{A}_i^*\boldsymbol{A}_i)\boldsymbol{x}-p^2\boldsymbol{A}_i^*\boldsymbol{b}_i.\end{aligned}$$

Letting $[\boldsymbol{A}_i^*\boldsymbol{A}_i]_{jk}$ denote the $(j,k)^{th}$ element of $\boldsymbol{A}_i^*\boldsymbol{A}_i$, step (i) uses the fact that,

$$\mathbb{E}_\delta[\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i] = \begin{cases} p[\boldsymbol{A}_i^*\boldsymbol{A}_i]_{jk}, & j = k, \\ p^2[\boldsymbol{A}_i^*\boldsymbol{A}_i]_{jk}, & j \neq k. \end{cases}$$

Now, we take the expectation with respect to the rows of $\boldsymbol{A}$ to obtain:

$$\begin{aligned} \mathbb{E}[g(\boldsymbol{x})] &\overset{(i)}{=} \frac{1}{p^2}\mathbb{E}\big[\tilde{\boldsymbol{A}}_i^*(\tilde{\boldsymbol{A}}_i\boldsymbol{x} - p\boldsymbol{b}_i)\big] - \frac{1-p}{p^2}\mathbb{E}\big[\mathrm{diag}(\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i)\big]\boldsymbol{x} \\ &= \frac{1}{p^2}\mathbb{E}\big[\nabla\tilde{f}_i(\boldsymbol{x})\big] - \frac{1-p}{p^2}\mathbb{E}\big[\mathrm{diag}(\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i)\big]\boldsymbol{x} \\ &\overset{(ii)}{=} \frac{1}{mp^2}\left(p^2\boldsymbol{A}^*\boldsymbol{A}\boldsymbol{x} + (p-p^2)\mathrm{diag}(\boldsymbol{A}^*\boldsymbol{A})\boldsymbol{x} - p^2\sum_i \boldsymbol{A}_i^*\boldsymbol{b}_i\right) - \frac{p(1-p)}{mp^2}\mathrm{diag}(\boldsymbol{A}^*\boldsymbol{A})\boldsymbol{x} \\ &= \frac{1}{m}\boldsymbol{A}^*\boldsymbol{A}\boldsymbol{x} + \frac{(p-p^2)}{mp^2}\mathrm{diag}(\boldsymbol{A}^*\boldsymbol{A})\boldsymbol{x} - \frac{1}{m}\boldsymbol{A}^*\boldsymbol{b} - \frac{p-p^2}{mp^2}\mathrm{diag}(\boldsymbol{A}^*\boldsymbol{A})\boldsymbol{x} \\ &= \frac{1}{m}\big(\boldsymbol{A}^*\boldsymbol{A}\boldsymbol{x} - \boldsymbol{A}^*\boldsymbol{b}\big) = \nabla F(\boldsymbol{x}). \end{aligned}$$

Step (i) follows from the definition of $g(\boldsymbol{x})$ and linearity of the expected value. Step (ii) utilizes (A.1) for the first expected value and evaluates the expectation of

$$\mathbb{E}[\mathrm{diag}(\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i)] = \mathbb{E}_i[\mathbb{E}_\delta[\mathrm{diag}(\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i)]] = p\mathbb{E}_i[\mathrm{diag}(\boldsymbol{A}_i^*\boldsymbol{A}_i)] = \frac{p}{m}\mathrm{diag}(\boldsymbol{A}^*\boldsymbol{A}).$$

The remaining steps follow by simplification. □

Lemma A.2. *The update function $g(\boldsymbol{x})$ of Algorithm* 2.1 *is Lipschitz continuous with Lipschitz constant $L_{i,D}$. In other words, for all $\boldsymbol{x},\boldsymbol{y} \in \mathscr{W}$,*

$$\|g(\boldsymbol{x}) - g(\boldsymbol{y})\| \leq L_{i,D}\|\boldsymbol{x} - \boldsymbol{y}\|.$$

In addition, we can bound the supremum Lipschitz constant, L_g by

$$L_g = \sup_{i,D} L_{g,i,D} \leq \frac{a_{max}^2}{p^2},$$

where $a_{max}^2 = \max_i \|\boldsymbol{A}_i\|^2$.

Proof. First we show that the Lipschitz constant $L_{i,D}$ of $g(\boldsymbol{x})$

$$\begin{aligned} \|g(\boldsymbol{x}) - g(\boldsymbol{y})\| &= \left\|\left(\frac{1}{p^2}\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i - \frac{(1-p)}{p^2}\mathrm{diag}(\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i)\right)(\boldsymbol{x} - \boldsymbol{y})\right\| \\ &\leq \left\|\frac{1}{p^2}\tilde{\boldsymbol{A}}_i\tilde{\boldsymbol{A}}_i^* - \frac{(1-p)}{p^2}\mathrm{diag}(\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i)\right\| \|\boldsymbol{x} - \boldsymbol{y}\| \leq \frac{1}{p^2}\left\|\tilde{\boldsymbol{A}}_i\right\|^2 \|\boldsymbol{x} - \boldsymbol{y}\|. \end{aligned}$$

The last step follows from Weyl's Inequality which allows us to bound

$$\left\|\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i-(1-p)\mathrm{diag}(\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i)\right\|\le\|\tilde{\boldsymbol{A}}_i\|^2.$$

Therefore we conclude that the Lipschitz constant of $g(\boldsymbol{x})$ is $L_{i,D}=\frac{1}{p^2}\|\tilde{\boldsymbol{A}}_i\|^2$.

To determine the supremum Lipschitz constant, we simply bound $L_{i,D}$ over all possible rows and all possible binary masks:

$$L_g=\sup_{i,D}L_{i,D}=\sup_{i,D}\frac{1}{p^2}\left\|\tilde{\boldsymbol{A}}_i\right\|^2\le\frac{1}{p^2}\sup_i\left\|\boldsymbol{A}_i\right\|^2\le\frac{a_{max}^2}{p^2},$$

where a_{max}^2 is a largest row norm of $\boldsymbol{A}$. Note that the supremum over all possible binary masks D occurs when D is the ones matrix. □

Lemma A.3. *We can uniformly bound the expected value of the magnitude of the update function in the following way. We have that* $E\|g(\boldsymbol{x})\|^2\le G$*, where*

$$G=\frac{2B}{mp^2}\left(1+\frac{(1-p)(2-p)}{p}\right)\sum_i\left\|\boldsymbol{A}_i\right\|^4+\frac{2}{mp^2}\sum_i\left\|\boldsymbol{A}_i\right\|^2|\boldsymbol{b}_i|^2,$$

where $B=\max_{\boldsymbol{x}\in\mathscr{W}}\|\boldsymbol{x}\|^2$*. In addition, we have that*

- *if* $\boldsymbol{A}\boldsymbol{x}_\star=\boldsymbol{b}$ *(the linear system is consistent) then*

$$G_\star=\frac{2(1-p)(2-p)}{mp^3}\|\boldsymbol{x}_\star\|^2\sum_i\left\|\boldsymbol{A}_i\right\|^4.$$

- *if* $\boldsymbol{A}\boldsymbol{x}_\star=\boldsymbol{b}+\boldsymbol{r}$ *(the linear system is inconsistent) then*

$$G_\star=\frac{2(1-p)(2-p)}{mp^3}\|\boldsymbol{x}_\star\|^2\sum_i\left\|\boldsymbol{A}_i\right\|^4+\frac{2}{mp^2}\sum_i\left\|\boldsymbol{A}_i\right\|^2\boldsymbol{r}_i^2.$$

G and $G_\star$ are also defined in (2.5) *and* (2.6) *respectively.*

Proof. We begin this proof by showing the upper bound of $\mathbb{E}\left[\|g(\boldsymbol{x})\|^2\right]$ for a general $\boldsymbol{x}$. From here, we obtain $G_\star$ by substituting $\boldsymbol{x}$ with $\boldsymbol{x}_\star$ and making the appropriate assumptions on the consistency of the linear system. To get the uniform upper bound over all $\boldsymbol{x}$, we isolate $\|\boldsymbol{x}\|^2$ and bound the norm by $B=\max_{\boldsymbol{x}\in\mathscr{W}}\|\boldsymbol{x}\|^2$. We have,

$$\begin{aligned}
\mathbb{E}[\|g(\boldsymbol{x})\|^2]&=\mathbb{E}\left[\left\|\frac{1}{p^2}\left(\tilde{\boldsymbol{A}}_i^*(\tilde{\boldsymbol{A}}_i\boldsymbol{x}-p\boldsymbol{b}_i)\right)-\frac{(1-p)}{p^2}\mathrm{diag}(\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i)\boldsymbol{x}\right\|^2\right]\\
&\overset{(i)}{\le}\frac{2}{p^4}\mathbb{E}\left[\left\|\tilde{\boldsymbol{A}}_i^*(\tilde{\boldsymbol{A}}_i\boldsymbol{x}-p\boldsymbol{b}_i)\right\|^2\right]+\frac{2(1-p)^2}{p^4}\mathbb{E}\left[\|\mathrm{diag}(\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i)\boldsymbol{x}\|^2\right]\\
&\overset{(ii)}{=}\frac{2}{p^4}\mathbb{E}\left[\|\tilde{\boldsymbol{A}}_i\|^2(\tilde{\boldsymbol{A}}_i\boldsymbol{x}-p\boldsymbol{b}_i)^2\right]+\frac{2(1-p)^2}{p^4}\mathbb{E}\left[\|\mathrm{diag}(\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i)\boldsymbol{x}\|^2\right]\\
&\overset{(iii)}{\le}\frac{2}{p^4}\mathbb{E}\left[\left\|\boldsymbol{A}_i\right\|^2(\tilde{\boldsymbol{A}}_i\boldsymbol{x}-p\boldsymbol{b}_i)^2\right]+\frac{2(1-p)^2}{p^4}\mathbb{E}\left[\|\mathrm{diag}(\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i)\boldsymbol{x}\|^2\right].
\end{aligned}$$

Step (i) follows by Jensen's inequality, step (ii) is simplification and uses the fact that $(\tilde{\boldsymbol{A}}_i\boldsymbol{x} - p\boldsymbol{b}_i)$ is scalar. Lastly, step (iii) bounds the magnitude of a row of $\boldsymbol{A}$ with missing data by the magnitude of a row of $\boldsymbol{A}$ without missing data (i.e. $\|\tilde{\boldsymbol{A}}_i\| = \|\boldsymbol{D}_i\boldsymbol{A}_i\| \le \|\boldsymbol{A}_i\|$ for all $\boldsymbol{D}_i$). From here, we use the fact that $\mathbb{E} = \mathbb{E}_i\mathbb{E}_\delta$ to obtain the following:

$$\begin{aligned} &\mathbb{E}[\|g(\boldsymbol{x})\|^2] \\ \le&\frac{2}{p^4}\mathbb{E}_i\left[\left\|\boldsymbol{A}_i\right\|^2\underbrace{\mathbb{E}_\delta[(\tilde{\boldsymbol{A}}_i\boldsymbol{x}-p\boldsymbol{b}_i)^2]}_{\text{(A)}}\right] + \frac{2(1-p)^2}{p^4}\mathbb{E}_i\left[\underbrace{\mathbb{E}_\delta\left[\|\text{diag}(\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i)\boldsymbol{x}\|^2\right]}_{\text{(B)}}\right]. \end{aligned} \tag{A.2}$$

Now, we will focus on the computation of $\mathbb{E}_\delta$. First, we compute (A). We have that,

$$\begin{aligned} &\mathbb{E}_\delta\left[(\tilde{\boldsymbol{A}}_i\boldsymbol{x}-p\boldsymbol{b}_i)^2\right] \\ =&\mathbb{E}_\delta\left[(\tilde{\boldsymbol{A}}_i\boldsymbol{x})^2\right] - 2p\mathbb{E}_\delta\left[\tilde{\boldsymbol{A}}_i\right]\boldsymbol{x}\boldsymbol{b}_i + p^2\boldsymbol{b}_i^2 = \mathbb{E}_\delta\left[\left(\sum_{j=1}^n\tilde{\boldsymbol{A}}_{ij}\boldsymbol{x}_j\right)^2\right] - 2p^2\boldsymbol{A}_i\boldsymbol{x}\boldsymbol{b}_i + p^2\boldsymbol{b}_i^2 \\ =&\mathbb{E}_\delta\left[\sum_{j=1}^n\tilde{\boldsymbol{A}}_{ij}^2\boldsymbol{x}_j^2 + 2\sum_{j=1}^n\sum_{k=1}^{j-1}\tilde{\boldsymbol{A}}_{ij}\tilde{\boldsymbol{A}}_{ik}\boldsymbol{x}_j\boldsymbol{x}_k\right] - 2p^2\boldsymbol{A}_i\boldsymbol{x}\boldsymbol{b}_i + p^2\boldsymbol{b}_i^2 \\ =&\left(p\sum_{j=1}^n\boldsymbol{A}_{ij}^2\boldsymbol{x}_j^2 + 2p^2\sum_{j=1}^n\sum_{k=1}^{j-1}\boldsymbol{A}_{ij}\boldsymbol{A}_{ik}\boldsymbol{x}_j\boldsymbol{x}_k\right) - 2p^2\boldsymbol{A}_i\boldsymbol{x}\boldsymbol{b}_i + p^2\boldsymbol{b}_i^2 \\ \overset{(i)}{=}&\left(p^2\sum_{j=1}^n\boldsymbol{A}_{ij}^2\boldsymbol{x}_j^2 + (p-p^2)\sum_{j=1}^n\boldsymbol{A}_{ij}^2\boldsymbol{x}_j^2 + 2p^2\sum_{j=1}^n\sum_{k=1}^{j-1}\boldsymbol{A}_{i,j}\boldsymbol{A}_{i,k}\boldsymbol{x}_j\boldsymbol{x}_k\right) - 2p^2\boldsymbol{A}_i\boldsymbol{x}\boldsymbol{b}_i + p^2\boldsymbol{b}_i^2 \\ =&p^2\left(\sum_{j=1}^n\boldsymbol{A}_{ij}^2\boldsymbol{x}_j^2 + 2\sum_{j=1}^n\sum_{k=1}^{j-1}\boldsymbol{A}_{ij}\boldsymbol{A}_{ik}\boldsymbol{x}_j\boldsymbol{x}_k - 2\boldsymbol{A}_i\boldsymbol{x}\boldsymbol{b}_i + \boldsymbol{b}_i^2\right) + (p-p^2)\left(\sum_{j=1}^n\boldsymbol{A}_{ij}^2\boldsymbol{x}_j^2\right) \\ =&p^2\left(\left(\sum_{j=1}^n\boldsymbol{A}_{ij}\boldsymbol{x}_j\right)^2 - 2\boldsymbol{A}_i\boldsymbol{x}\boldsymbol{b}_i + \boldsymbol{b}_i^2\right) + (p-p^2)\boldsymbol{x}^*\text{diag}(\boldsymbol{A}_i^*\boldsymbol{A}_i)\boldsymbol{x} \\ =&p^2\left(\boldsymbol{A}_i\boldsymbol{x}-\boldsymbol{b}_i\right)^2 + p(1-p)\boldsymbol{x}^*\text{diag}(\boldsymbol{A}_i^*\boldsymbol{A}_i)\boldsymbol{x}. \end{aligned}$$

In step (i), we add and subtract the term $p^2\sum_{j=1}^n\boldsymbol{A}_{i,j}^2\boldsymbol{x}_j^2$ so that we can combine terms. Other equalities follow by simplification and computation of expected value. Note that

$$\mathbb{E}_\delta[\tilde{\boldsymbol{A}}_{i,j}] = p\boldsymbol{A}_{i,j}, \qquad \mathbb{E}_\delta[\tilde{\boldsymbol{A}}_{i,j}\tilde{\boldsymbol{A}}_{i,k}] = p^2\boldsymbol{A}_{i,j}\boldsymbol{A}_{i,k}, \qquad if\, j \neq k.$$

For term (B), we simply compute that

$$\begin{aligned}&\mathbb{E}_\delta\left[\|\mathrm{diag}(\tilde{\boldsymbol{A}}_i^*\tilde{\boldsymbol{A}}_i)\boldsymbol{x}\|^2\right]\\=&\mathbb{E}_\delta\left[\sum_{j=1}^n\tilde{\boldsymbol{A}}_{i,j}^2\boldsymbol{x}_j^2\right]=p\sum_{j=1}^n(\boldsymbol{A}_{i,j}^2\boldsymbol{x}_j^2)\ =p\left\|\mathrm{diag}(\boldsymbol{A}_i^*\boldsymbol{A}_i)\boldsymbol{x}\right\|^2.\end{aligned}$$

Now that we have (A) and (B), we can compute a general upper bound for $\mathbb{E}[\|g(\boldsymbol{x})\|^2]$. Starting with substituting (A) and (B) into (A.2),

$$\begin{aligned}\mathbb{E}\left[\|g(\boldsymbol{x})\|^2\right]&\overset{(i)}{\leq}\frac{2}{p^2}\mathbb{E}_i\left[\left\|\boldsymbol{A}_i\right\|^2\left(\boldsymbol{A}_i\boldsymbol{x}-\boldsymbol{b}_i\right)^2\right]+\frac{2p(1-p)}{p^4}\mathbb{E}_i\left[\left\|\boldsymbol{A}_i\right\|^2\boldsymbol{x}^*\mathrm{diag}(\boldsymbol{A}_i^*\boldsymbol{A}_i)\boldsymbol{x}\right]\\&\qquad+\frac{2p(1-p)^2}{p^4}\mathbb{E}_i\left[\left\|\mathrm{diag}(\boldsymbol{A}_i^*\boldsymbol{A}_i)\boldsymbol{x}\right\|^2\right]\\&\overset{(ii)}{\leq}\frac{2}{p^2}\mathbb{E}_i\left[\left\|\boldsymbol{A}_i\right\|^2\left(\boldsymbol{A}_i\boldsymbol{x}-\boldsymbol{b}_i\right)^2\right]+\left(\frac{2p(1-p)}{p^4}+\frac{2p(1-p)^2}{p^4}\right)\mathbb{E}_i\left[\left\|\boldsymbol{A}_i\right\|^2\boldsymbol{x}^*\mathrm{diag}(\boldsymbol{A}_i^*\boldsymbol{A}_i)\boldsymbol{x}\right]\\&\leq\frac{2}{p^2}\mathbb{E}_i\left[\left\|\boldsymbol{A}_i\right\|^2\left(\boldsymbol{A}_i\boldsymbol{x}-\boldsymbol{b}_i\right)^2\right]+\frac{2p(1-p)(2-p)}{p^4}\mathbb{E}_i\left[\left\|\boldsymbol{A}_i\right\|^2\boldsymbol{x}^*\mathrm{diag}(\boldsymbol{A}_i^*\boldsymbol{A}_i)\boldsymbol{x}\right]\\&\overset{(iii)}{=}\frac{2}{mp^2}\sum_i\left\|\boldsymbol{A}_i\right\|^2\left(\boldsymbol{A}_i\boldsymbol{x}-\boldsymbol{b}_i\right)^2+\frac{2p(1-p)(2-p)}{mp^4}\sum_i\left\|\boldsymbol{A}_i\right\|^2\boldsymbol{x}^*\mathrm{diag}(\boldsymbol{A}_i^*\boldsymbol{A}_i)\boldsymbol{x}\\&\overset{(iv)}{\leq}\frac{2}{mp^2}\sum_i\left\|\boldsymbol{A}_i\right\|^2\left(\boldsymbol{A}_i\boldsymbol{x}-\boldsymbol{b}_i\right)^2+\frac{2p(1-p)(2-p)}{mp^4}\|\boldsymbol{x}\|^2\sum_i\left\|\boldsymbol{A}_i\right\|^4.\end{aligned}$$

Step (i) substitutes (A) and (B) in (A.2). Step (ii) uses the fact that

$$\left\|\mathrm{diag}(\boldsymbol{A}_i^*\boldsymbol{A}_i)\boldsymbol{x}\right\|^2\leq\left\|\mathrm{diag}(\boldsymbol{A}_i)\mathrm{diag}(\boldsymbol{A}_i)\boldsymbol{x}\right\|^2\leq\left\|\boldsymbol{A}_i\right\|^4\|\boldsymbol{x}\|^2.$$

From here, we substitute $\boldsymbol{x}$ with $\boldsymbol{x}_\star$ to compute $G_\star$. If $\boldsymbol{A}\boldsymbol{x}_\star=\boldsymbol{b}$ (the linear system is consistent) then the terms $\left(\boldsymbol{A}_i\boldsymbol{x}-\boldsymbol{b}_i\right)^2=0$ and we find that

$$G_\star=\frac{2(1-p)(2-p)}{mp^3}\|\boldsymbol{x}_\star\|^2\sum_i\left\|\boldsymbol{A}_i\right\|^4.$$

Otherwise, if $\boldsymbol{A}\boldsymbol{x}_\star=\boldsymbol{b}+\boldsymbol{r}$ for some residual vector $\boldsymbol{r}$, we have that

$$G_\star=\frac{2}{mp^2}\sum_i\left\|\boldsymbol{A}_i\right\|^2\boldsymbol{r}_i^2+\frac{2(1-p)(2-p)}{mp^3}\|\boldsymbol{x}_\star\|^2\sum_i\left\|\boldsymbol{A}_i\right\|^4,$$

where $\boldsymbol{r}_i$ is the i^{th} element of the vector $\boldsymbol{r}$. To finish the proof of Lemma A.3, we simplify starting from step (iv).

$$\begin{aligned}
\mathbb{E}\left[\|g(\boldsymbol{x})\|^2\right] &\leq \frac{2}{mp^2}\sum_i \left\|\boldsymbol{A}_i\right\|^2\left(\boldsymbol{A}_i\boldsymbol{x}-\boldsymbol{b}_i\right)^2+\frac{2p(1-p)(2-p)}{mp^4}\|\boldsymbol{x}\|^2\sum_i\left\|\boldsymbol{A}_i\right\|^4\\
&\overset{(i)}{\leq}\frac{2}{mp^2}\sum_i\left\|\boldsymbol{A}_i\right\|^2\left(|\boldsymbol{A}_i\boldsymbol{x}|^2+|\boldsymbol{b}_i|^2\right)+\frac{2p(1-p)(2-p)}{mp^4}\|\boldsymbol{x}\|^2\sum_i\left\|\boldsymbol{A}_i\right\|^4\\
&\overset{(ii)}{\leq}\frac{2}{mp^2}\sum_i\left\|\boldsymbol{A}_i\right\|^4\|\boldsymbol{x}\|^2+\frac{2}{mp^2}\sum_i\left\|\boldsymbol{A}_i\right\|^2|\boldsymbol{b}_i|^2+\frac{2p(1-p)(2-p)}{mp^4}\|\boldsymbol{x}\|^2\sum_i\left\|\boldsymbol{A}_i\right\|^4\\
&\overset{(iii)}{\leq}\frac{2}{mp^2}\sum_i\left\|\boldsymbol{A}_i\right\|^4B+\frac{2}{mp^2}\sum_i\left\|\boldsymbol{A}_i\right\|^2|\boldsymbol{b}_i|^2+\frac{2p(1-p)(2-p)}{mp^4}B\sum_i\left\|\boldsymbol{A}_i\right\|^4\\
&=\left(\frac{2B}{mp^2}+\frac{2p(1-p)(2-p)B}{mp^4}\right)\sum_i\left\|\boldsymbol{A}_i\right\|^4+\frac{2}{mp^2}\sum_i\left\|\boldsymbol{A}_i\right\|^2|\boldsymbol{b}_i|^2\\
&=\frac{2B}{mp^2}\left(1+\frac{p(1-p)(2-p)}{p^2}\right)\sum_i\left\|\boldsymbol{A}_i\right\|^4+\frac{2}{mp^2}\sum_i\left\|\boldsymbol{A}_i\right\|^2|\boldsymbol{b}_i|^2\\
&=\frac{2B}{mp^2}\left(1+\frac{(1-p)(2-p)}{p}\right)\sum_i\left\|\boldsymbol{A}_i\right\|^4+\frac{2}{mp^2}\sum_i\left\|\boldsymbol{A}_i\right\|^2|\boldsymbol{b}_i|^2
\end{aligned}$$

In step (i) we use Jensen's inequality. Note that $\boldsymbol{A}_i\boldsymbol{x}$ and $\boldsymbol{b}_i$ are both scalar values. In step (ii), we distribution the summation in the first term and use the fact that $|\boldsymbol{A}_i\boldsymbol{x}|^2 \leq \|\boldsymbol{A}_i\|^2\|\boldsymbol{x}\|^2$ by the Cauchy-Schwarz inequality. Step (iii) uses the definition of $B=\max_{\boldsymbol{x}\in\mathscr{W}}\|\boldsymbol{x}\|^2$. The remaining lines are simplification. □

Before we begin the proof of Theorem 2.2, we remind the reader that $F(\boldsymbol{x})$ is strongly convex with strong convexity parameter μ. In other words, for all $\boldsymbol{x},\boldsymbol{y}\in\mathscr{W}$ we have that

$$(\boldsymbol{x}-\boldsymbol{y})^*(\nabla F(\boldsymbol{x})-\nabla F(\boldsymbol{y}))\geq\mu\|\boldsymbol{x}-\boldsymbol{y}\|^2. \tag{A.3}$$

In addition, we define a new function

$$G(\boldsymbol{x})=\frac{1}{2p^2}\left((\tilde{\boldsymbol{A}}_i\boldsymbol{x}-p\boldsymbol{b}_i)^2-\frac{(1-p)}{2p^2}\|\text{diag}(\tilde{\boldsymbol{A}}_i)\boldsymbol{x}\|^2\right)$$

so that $g(\boldsymbol{x})=\nabla G(\boldsymbol{x})$. The update function $g(\boldsymbol{x})$ follows the Co-coercivity Lemma as stated in Lemma A.4.

Lemma A.4. ([18], Lemma A.1) *For $G(\boldsymbol{x})$ a smooth function such that $\nabla G(\boldsymbol{x})=g(\boldsymbol{x})$,*

$$\|g(\boldsymbol{x})-g(\boldsymbol{y})\|^2\leq L_{i,D}(\boldsymbol{x}-\boldsymbol{y})^*(g(\boldsymbol{x})-g(\boldsymbol{y})),$$

where $g(\boldsymbol{x})$ has Lipschitz constant $L_{i,D}$.

A.1. Proof of theorem 2.2

Proof. First, we bound expected error conditional on the previous $k-1$ iterations. Let $\mathbb{E}_{k-1}[\cdot]$ denote the expected value conditional of the previous $k-1$ iterations and note that by the Law of Iterated Expectation, we have that the full expected value over all iterations is $\mathbb{E}[\cdot] = \mathbb{E}[\mathbb{E}_{k-1}[\cdot]]$. Thus,

$$\begin{aligned}
&\mathbb{E}_{k-1}\left[\|\boldsymbol{x}_k - \boldsymbol{x}_\star\|^2\right] = \mathbb{E}_{k-1}\left[\|\boldsymbol{x}_{k-1} - \alpha g(\boldsymbol{x}_{k-1}) - \boldsymbol{x}_\star\|^2\right] \\
&\overset{(i)}{=} \|\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star\|^2 - 2\alpha(\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star)^*\mathbb{E}_{k-1}\left[g(\boldsymbol{x}_{k-1})\right] + \alpha^2\mathbb{E}_{k-1}\left[\|g(\boldsymbol{x}_{k-1})\|^2\right] \\
&\overset{(ii)}{=} \|\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star\|^2 - 2\alpha(\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star)^*(\nabla F(\boldsymbol{x}_{k-1}) - \nabla F(\boldsymbol{x}_\star)) + \alpha^2\mathbb{E}_{k-1}\left[\|g(\boldsymbol{x}_{k-1})\|^2\right] \\
&\overset{(iii)}{\leq} \|\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star\|^2 - 2\alpha(\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star)^*(\nabla F(\boldsymbol{x}_{k-1}) - \nabla F(\boldsymbol{x}_\star)) \\
&\qquad\qquad + 2\alpha^2\mathbb{E}_{k-1}\Big\|g(\boldsymbol{x}_{k-1}) - g(\boldsymbol{x}_\star)\|^2\Big] + \alpha^2\mathbb{E}_{k-1}\left[\|g(\boldsymbol{x}_\star)\|^2\right] \\
&\overset{(iv)}{\leq} \|\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star\|^2 - 2\alpha(\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star)^*(\nabla F(\boldsymbol{x}_{k-1}) - \nabla F(\boldsymbol{x}_\star)) \\
&\qquad\qquad + 2\alpha^2 L_{i,g}(\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star)^*(\mathbb{E}_{k-1}[g(\boldsymbol{x}_{k-1})] - \mathbb{E}_{k-1}[g(\boldsymbol{x}_\star)]) + \alpha^2 G_\star \\
&\overset{(v)}{\leq} \|\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star\|^2 - 2\alpha(\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star)^*(\nabla F(\boldsymbol{x}_{k-1}) - \nabla F(\boldsymbol{x}_\star)) \\
&\qquad\qquad + 2\alpha^2 L_g(\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star)^*(\nabla F(\boldsymbol{x}_{k-1}) - \nabla F(\boldsymbol{x}_\star)) + \alpha^2 G_\star \\
&\leq \|\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star\|^2 - 2\alpha(1 - \alpha L_g)(\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star)^*(\nabla F(\boldsymbol{x}_{k-1}) - \nabla F(\boldsymbol{x}_\star)) + \alpha^2 G_\star \\
&\overset{(vi)}{\leq} \|\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star\|^2 - 2\alpha(1 - \alpha L_g)\mu\|\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star\|^2 + \alpha^2 G_\star \\
&= \big(1 - 2\alpha\mu(1 - \alpha L_g)\big)\|\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star\|^2 + \alpha^2 G_\star \\
&= r\|\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star\|^2 + \alpha^2 G_\star.
\end{aligned}$$

Step (i) follows from the definition of the ℓ_2 norm. Step (ii) takes the expectation of $g(\boldsymbol{x}_{k-1})$ using Lemma A.1 and uses the fact that $\nabla F(\boldsymbol{x}_\star) = 0$ to subtract $2\alpha(\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star)^*\nabla F(\boldsymbol{x}_\star)$. In step (iii) we add and subtract the term $\|g(\boldsymbol{x}_\star)\|^2$ then apply Jensen's inequality. Step (iv) is an application of the Lemma A.4. Step (v) bounds $L_{i,D}$ by $L_g = \sup_{i,D} L_{i,D}$ and uses Lemma A.1 to compute the expectation of $\mathbb{E}_{k-1}[g(\boldsymbol{x})]$. We use the strong convexity of $F(\boldsymbol{x})$ in step (vi). The remaining lines are simplification. Now, by the Law of Iterated Expectation we recursively apply this bound to obtain the desired result,

$$\begin{aligned}
&\mathbb{E}\|\boldsymbol{x}_k - \boldsymbol{x}_\star\|^2 \leq r\mathbb{E}_{k-2}\|\boldsymbol{x}_{k-1} - \boldsymbol{x}_\star\|^2 + \alpha^2 G_\star \\
\leq& r^k\|\boldsymbol{x}_0 - \boldsymbol{x}_\star\|^2 + \alpha^2 G_\star\sum_{j=0}^{k-1} r^j \leq r^k\|\boldsymbol{x}_0 - \boldsymbol{x}_\star\|^2 + \frac{\alpha^2 G_\star}{1-r}.
\end{aligned}$$

This completes the proof of Theorem 2.2. □

A note on Inconsistent Linear Systems. Theorem 2.2 also applies to inconsistent systems. Let $\boldsymbol{A}\boldsymbol{x}_\star = \boldsymbol{b} + \boldsymbol{r}$ where $\boldsymbol{r} \in \text{null}(\boldsymbol{A}^*)$. In the proof of Theorem 2.2, we use the fact that $\nabla F(\boldsymbol{x}_\star) = 0$ in step (ii). This is still true in the inconsistent setting as $\nabla F(\boldsymbol{x}_\star) = \boldsymbol{A}^*(\boldsymbol{A}\boldsymbol{x}_\star - \boldsymbol{b}) = \boldsymbol{A}^*\boldsymbol{r} = 0$. All other computations go through without issue.

Acknowledgments Needell was partially supported by NSF CAREER Grant No. 1348721, NSF BIGDATA 1740325, and the Alfred P. Sloan Fellowship. Ma was supported in part by NSF CAREER Grant No. 1348721, the CSRC Intellisis Fellowship, and the Edison International Scholarship.

References

[1] L. Bottou, *Large-scale machine learning with stochastic gradient descent*, Proc. of COMPSTAT'2010, Springer, 2010, pp. 177–186.

[2] L. Bottou, *Stochastic gradient descent tricks*, *Neural Networks: Tricks of the Trade*, Springer, 2012, pp. 421–436.

[3] J. F. Cai, E. J. Candès and Z. Shen, *A singular value thresholding algorithm for matrix completion*, SIAM J. Optimz., 20(4) (2010), pp. 1956–1982.

[4] Y. Censor, P. P. Eggermont and D. Gordon, *Strong underrelaxation in Kaczmarz's method for inconsistent systems*, Numer. Math., 41(1) (1983), pp. 83–92.

[5] A. P. Dempster, N. M. Laird and D. B. Rubin, *Maximum likelihood from incomplete data via the em algorithm*, J. Roy. Stat. Soc. B Met., 1977, pp. 1–38.

[6] B. Efron, *Missing data, imputation, and the bootstrap*, J. Am. Stat. Assoc., 89(426) (1994), pp. 463–475.

[7] M. Fichman and J. N. Cummings, *Multiple imputation for missing data: Making the most of what you know*, Organ. Res. Methods, 6(3) (2003), pp. 282–308.

[8] D. Goldberg, D. Nichols, B. M. Oki and D. Terry, *Using collaborative filtering to weave an information tapestry*, Communications of the ACM, 35(12) (1992), pp. 61–70.

[9] M. Hanke and W. Niethammer, *On the acceleration of Kaczmarz's method for inconsistent linear systems*, Linear Algebra Appl., 130 (1990), pp. 83–98.

[10] S. Kaczmarz, *Angenäherte auflösung von systemen linearer gleichungen*, Bull. Int. Acad. Polon. Sci. Lett. Ser. A, 35 (1937), pp. 355–357.

[11] R. H. Keshavan, A. Montanari and S. Oh, *Matrix completion from noisy entries*, J. Machine Learning Research, 11(2010), pp. 2057–2078.

[12] R. H. Keshavan, S. Oh and A. Montanari, *Matrix completion from a few entries*, IEEE T. Inform. Theory, 2009, pp. 324–328.

[13] D. Leventhal and A. S. Lewis, *Randomized methods for linear constraints: convergence rates and conditioning*, Math. Oper. Res., 35(3) (2010), pp. 641–654.

[14] M. Lichman, UCI Machine Learning Repository, 2013.

[15] R. J. Little and D. B. Rubin, Statistical Analysis with Missing Data, John Wiley & Sons, 2014.

[16] A. Ma, D. Needell and A. Ramdas, *Convergence properties of the randomized extended gauss–seidel and kaczmarz methods*, SIAM J. Matrix Anal. Appl., 36(4) (2015), pp. 1590–1604.

[17] B. M. Marlin, R. S. Zemel, S. Roweis and M. Slaney, *Collaborative filtering and the missing at random assumption*, the Twenty-Third Conference on Uncertainty in Artificial Intelligence, AUAI Press, 2007, pp. 267–275.

[18] D. Needell, R. Ward and N. Srebro, *Stochastic gradient descent, weighted sampling, and the randomized kaczmarz algorithm*, Proc. Adv. in Neural Processing Systems (NIPS), 2014, pp. 1017–1025.

[19] B. Recht, *A simpler approach to matrix completion*, J. Machine Learning Research, 12 (2011), pp. 3413–3430.

[20] H. Robbins and S. Monro, *A stochastic approximation method*, Ann. Math. Statist., 1951, pp. 400–407.

[21] M. Schmidt and N. L. Roux, Fast Convergence of Stochastic Gradient Descent under a Strong Growth Condition, Mathematics, 2013.

[22] O. Shamir and T. Zhang, *Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes*, Proc. Int. Conf. Machine Learning, 2013, pp. 71–79.

[23] T. Strohmer and R. Vershynin, *A randomized kaczmarz algorithm with exponential convergence*, J. Fourier Anal. Appl., 15(2) (2009), pp. 262–278.

[24] T. Zhang, *Solving large scale linear prediction problems using stochastic gradient descent algorithms*, Proc. Int. Conf. Machine Learning, ACM, 2004, pp. 116.