11-4-2016

A Sampling Kaczmarz-Motzkin Algorithm for Linear Feasibility

Jesus A. De Loera
University of California - Davis

Jamie Haddock
University of California, Davis

Deanna Needell
Claremont McKenna College

A Sampling Kaczmarz-Motzkin Algorithm for Linear Feasibility

Jesús A. De Loera, Jamie Haddock, Deanna Needell

Abstract

We combine two iterative algorithms for solving large-scale systems of linear inequalities, the relaxation method of Agmon, Motzkin et al. and the randomized Kaczmarz method. We obtain a family of algorithms that generalize and extend both projection-based techniques. We prove several convergence results, and our computational experiments show our algorithms often outperform the original methods.

1 Introduction

We are interested solving large-scale systems of linear inequalities $Ax \leq b$. Here $b \in \mathbb{R}^m$ and A an $m \times n$ matrix; the regime $m \gg n$ is our setting of interest, where iterative methods are typically employed. We denote the rows of A by the vectors $a_1, a_2, \ldots, a_m$. It is an elementary fact that the set of all $x \in \mathbb{R}^n$ that satisfy the above constraints is a convex polyhedral region, which we will denote by P. This paper merges two iterated-projection methods, the relaxation method of Agmon, Motzkin et al. and the randomized Kaczmarz method. For the most part, these two methods have not met each other and have not been analyzed in a unified framework. The combination of these two algorithmic branches of thought results in an interesting new family of algorithms which generalizes and outperforms its predecessors. We begin with a short description of these two classical methods.

Motzkin's method. The first branch of research in linear feasibility is the so-called *relaxation method* or *Motzkin's method*. It is clear from the literature that this is not well-known, say among researchers in machine learning, and some results have been re-discovered several times. E.g., the famous 1958 *perceptron* algorithm [Ros58] can be thought of a member of this family of methods; but the very first relaxation-type algorithm analysis appeared a few years earlier in 1954, within the work of Agmon [Agm54], and Motzkin and Schoenberg [MS54]. Additionally, the relaxation method has been referred to as the Kaczmarz method with the "most violated constraint control" or the "maximal-residual control" [Cen81, NSV$^+$16, PP15]. This method can be described as follows: Starting from any initial point x_0, a sequence of points is generated. If the current point x_i is feasible we stop, else there must be a constraint $a^T x \leq b$ that is *most violated*. The constraint defines a hyperplane H. If w_H is the orthogonal projection of x_i onto the hyperplane H, choose a number λ (normally chosen between 0 and 2), and the new point x_{i+1} is given by $x_{i+1} = x_i + \lambda(w_H - x_i)$. Figure 1 displays the iteration visually.

Many modifications and analyses of this technique have been published since the 1950s, creating an extensive bibliography. For example, versions of the relaxation method have suggested various choices of step-length multiplier, λ (throughout this paper we consider $\lambda \in (0, 2]$), and various choices for the violated hyperplane. The rate of convergence of Motzkin's method depends not only on λ, but also on the *Hoffman constants* investigated first by Agmon [Agm54] and then later by Hoffmann [Hof52]. If the system of inequalities $Ax \leq b$ is feasible, i.e. $P \neq \emptyset$, then there exists Hoffman constants L_∞ and L_2 so that $d(x, P) \leq L_\infty \|(Ax - b)^+\|_\infty$ and $d(x, P) \leq L_2 \|(Ax - b)^+\|_2$ for all x (here and throughout, z^+ denotes the positive entries of the vector z with zeros elsewhere and $d(x, P)$ the usual distance between a point x

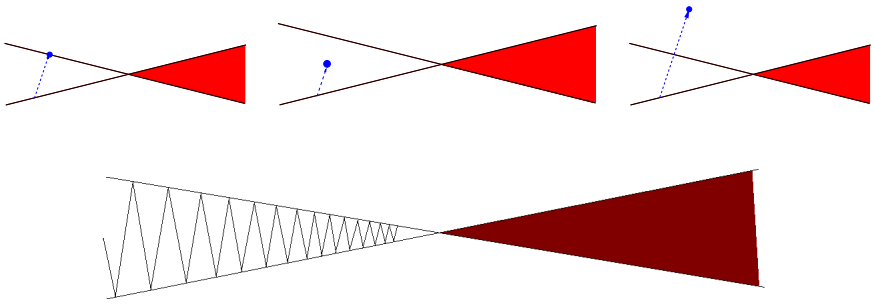

Figure 1: three projections with $\lambda = 1$, $\lambda < 1$ and $\lambda > 1$ and a visualization of several steps of the algorithm.

and the polytope P). The constants satisfy $L_\infty \leq \sqrt{m} L_2$. When the system of inequalities $Ax \leq b$ defines a consistent system of equations $\tilde{A}x = \tilde{b}$ with full column-rank matrix $\tilde{A}$, then the Hoffman constant is simply the norm of the left inverse, $\|\tilde{A}^{-1}\|_2$. With these constants, one can prove convergence rate results like the following (a spin-off of Theorem 3 of [Agm54] which is easily proven in the style of [LL10]):

Proposition 1. Consider a normalized system with $\|a_i\| = 1$ for all $i = 1, ..., m$. If the feasible region P is nonempty then the relaxation method converges linearly:

$$d(x_k, P)^2 \leq \left(1 - \frac{2\lambda - \lambda^2}{L_\infty^2}\right)^k d(x_0, P)^2 \leq \left(1 - \frac{2\lambda - \lambda^2}{mL_2^2}\right)^k d(x_0, P)^2.$$

A bad feature of the standard version of the relaxation method using real-valued data is that when the system $Ax \leq b$ is infeasible it cannot terminate, as there will always be a violated inequality. In the 1980's the relaxation method was revisited with interest because of its similarities to the ellipsoid method (see [AH05, Bet04, Gof80, Tel82] and references therein). One can show that the relaxation method is finite in all cases when using rational data, in that it can be modified to detect infeasible systems. In some special cases the method gives a polynomial time algorithm (e.g. for totally unimodular matrices [MTA81]), but there are also examples of exponential running times (see [Gof82, Tel82]). In late 2010, Chubanov [Chu12], announced a modification of the traditional relaxation style method, which gives a *strongly polynomial*-time algorithm in some situations [BDJ14, VZ14]. Unlike [Agm54, MS54], who only projected onto the original hyperplanes that describe the polyhedron P, Chubanov [Chu12] projects onto new, auxiliary inequalities which are linear combinations of the input. See Figure 2 for an example of this process.

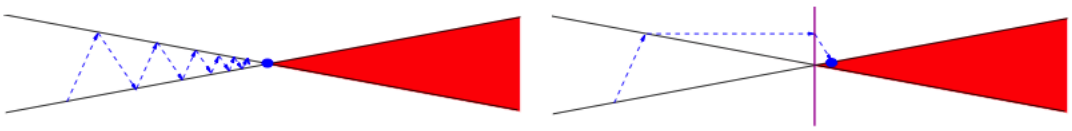

Figure 2: Left: Projecting onto original hyperplanes. Right: Projecting onto an induced hyperplane (like those in Chubanov's method).

Kaczmarz method. The second research branch is that of the Kaczmarz method [Kac37, GBH70] which is one of the most popular solvers of overdetermined systems of linear equations due to its speed and simplicity. Just like Motzkin's, it is an iterative method which consists of a series of alternating orthogonal

projections onto the hyperplanes defined by the system of equations. The original Kaczmarz method simply cycles through the equations sequentially, so its convergence rate depends on the order of the rows. One way to overcome this is to use the equations in a *random order*, rather than sequentially [HS78, HM93, Nat01]. More precisely, we begin with $Ax \leq b$, a linear system of inequalities where A is an $m \times n$ matrix with rows a_i and x_0 an initial guess. For $k = 0, 1, 2, ...$ one defines

$$x_{k+1} = x_k - \frac{(\langle a_i, x_k \rangle - b_i)^+}{\|a_i\|_2^2} a_i$$

where i is chosen from $\{1, 2, ..., m\}$ at random, say with probability proportional to $\|a_i\|_2^2$. Thus, x_k is the projection of x_{k-1} onto the hyperplane $\{x | a_i^T x = b_i\}$. Strohmer and Vershynin [SV09] provided an elegant convergence analysis of the randomized Kaczmarz method for consistent equations. Later, Leventhal and Lewis [LL10] extended the probabilistic analysis from systems of equations to systems of linear inequalities. They focused on giving bounds on the convergence rate that take into account the numerical conditions captured by the Hoffman constants L_∞ and L_2. If one additionally makes use of a projection parameter, $\lambda \neq 1$, you can easily extend the convergence rate in [LL10] to account for this:

Proposition 2. If the feasible region, P, is nonempty then the Randomized Kaczmarz method with projection parameter λ converges linearly in expectation:

$$\mathbb{E}[d(x_k, P)^2] \leq \left(1 - \frac{2\lambda - \lambda^2}{\|A\|_F^2 L_2^2}\right)^k d(x_0, P)^2.$$

Note the similarities between Propositions 1 and 2: the convergence rate constants are identical for normalized systems ($\|A\|_F^2 = m$).

The work of Strohmer and Vershynin sparked a new interest in the Kaczmarz approach and there have been many recent developments in the method and its analysis. Needell [Nee10] extended this work to the case of inconsistent systems of equations, showing exponential convergence down to some fixed *convergence horizon*, see also [WAL15]. In order to break this convergence horizon, one needs to modify the Kaczmarz method since by design it projects exactly onto a given hyperplane. Zouzias and Freris [ZF12] analyzed an extended randomized Kaczmarz method which incorporates an additional projection step to reduce the size of the residual. This was extended to the block case in [NZZ15]. The relation of these approaches to coordinate descent and gradient descent methods has also been recently studied, see e.g. [GO12, Dum14, NSW14a, OZ15a, MNR15, HNR15, OZ15a, GR15].

Other variations to the Kaczmarz method include block methods [Elf80, EHL81, NW13, NT13, BN, XZ02] which have been shown to offer acceleration for certain systems of equations with fast-multipliers. Other acceleration and convergence schemes focus on sampling selections [AWL14,EN11,NSW14b,OZ15b], projection parameters [WM67,CEG83,Tan71,HN90], adding row directions [PPKR12], parallelized implementations [LWS14, ADG14], structure exploiting approaches [LW15, LMY15], and the use of preconditioning [GPS16]. Some other references on recent work include [CP12, RM12]

For the most part, it seems that these two branches of research which address the same problems have been developing disjointly from each other. For example, the idea of taking linear combinations of the constraints was first exploited in [Chu12], but was recently re-discovered and reproduced for linear equations in [GR15], but the authors seem unaware of the optimizers work in the more general setting of linear inequalities in [Chu12, BDJ14, VZ14]. Another example is the manipulation of the projection parameter λ [WM67, CEG83, Tan71, HN90]. It is a goal of this paper to bridge the separation between these two branches of research that essentially study the same iterative projection procedure. In this paper we explore a family of hybrid algorithms that use elements from both groups of research.

1.1 Our contribution: the Sampling Kaczmarz-Motzkin method

Despite the similarity between the Kaczmarz and Motzkin methods (the difference only being in the selection criterion), work on these approaches has remained for the most disjoint. Our proposed family of methods, which we refer to as the *Sampling Kaczmarz-Motzkin* (SKM) methods, are intended to balance the pros and cons of these related methods. Namely, the relaxation method forms iterates whose distance to the polyhedral solution space are monotonically decreasing; however, the time required to choose the most violated hyperplane in each iteration is costly. Conversely, the Randomized Kaczmarz method has a very inexpensive cost per iteration; however, the method has slow convergence when many of the constraints are satisfied. Our methods will still have a probabilistic choice, like in randomized Kaczmarz, but make strong use of the maximum violation criterion within this random sample of the constraints. Our method is easily seen to interpolate between what was proposed in [LL10] and in [MS54].

Method (SKM method)**.** Suppose $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$. Let $x_0 \in \mathbb{R}^n$ be given. Fix $0 < \lambda \leq 2$. We iteratively construct approximations to a solution lying in P in the following way:

1. Choose a sample of β constraints, τ_k, uniformly at random from among the rows of A.
2. From among these β constraints, choose $t_k := \underset{i\in\tau_k}{\operatorname{argmax}}\ a_i^T x_{k-1} - b_i$.
3. Define $x_k := x_{k-1} - \lambda \frac{(a_{t_k}^T x_{k-1} - b_{t_k})^+}{\|a_{t_k}\|^2} a_{t_k}$.
4. Repeat.

Remark: the SKM method with $\beta = m$ recovers the Motzkin relaxation methods, while the SKM method with $\beta = 1$ gives a variant of the randomized Kaczmarz method. We now state our first main result.

Theorem 1. Let A be normalized so $\|a_i\|^2 = 1$ for all rows i. If the feasible region P is nonempty then the SKM method with samples of size β converges at least linearly in expectation and the bound on the rate depends on the number of satisfied constraints in the system $Ax \leq b$. More precisely, let s_{k-1} be the number of satisfied constraints after iteration $k-1$ and $V_{k-1} = \max\{m - s_{k-1}, m - \beta + 1\}$; then, in the kth iteration,

$$\mathbb{E}[d(x_k, P)^2] \leq \left(1 - \frac{2\lambda - \lambda^2}{V_{k-1}L_2^2}\right) d(x_{k-1}, P)^2 \leq \left(1 - \frac{2\lambda - \lambda^2}{mL_2^2}\right)^k d(x_0, P)^2.$$

Our second main theoretical result notes that, for rational data, one can provide a *certificate of feasibility* after finitely many iterations of SKM. This is an extension of the results by Telgen [Tel82] who also noted the connection between relaxation techniques and the ellipsoid method. To explain what we mean by a certificate of feasibility we recall the length of the binary encoding of a linear feasibility problem with rational data is

$$\sigma = \sum_i \sum_j \log(|a_{ij}| + 1) + \sum_i \log(|b_i| + 1) + \log nm + 2.$$

Denote the maximum violation of a point $x \in \mathbb{R}^n$ as $\theta(x) = \max\{0, \max_i\{a_i^T x - b_i\}\}$.

Telgen's proof of the finiteness of the relaxation method makes use of the following lemma (which is key in demonstrating that Khachian's ellipsoidal algorithm is finite and polynomial-time [Hač79]):

Lemma 1. If the rational system $Ax \leq b$ is infeasible, then for all $x \in \mathbb{R}^n$, the maximum violation satisfies $\theta(x) \geq 2 * 2^{-\sigma}$.

Thus, to detect feasibility of the rational system $Ax \leq b$, we need only find a point, x_k with $\theta(x_k) < 2 * 2^{-\sigma}$; such a point will be called a *certificate of feasibility*.

In the following theorem, we demonstrate that we expect to find a certificate of feasibility, when the system is feasible, and that if we do not find a certificate after finitely many iterations, we can put a lower bound on the probability that the system is infeasible. Furthermore, if the system is feasible, we can bound the probability of finding a certificate of feasibility.

Theorem 2. Suppose A, b are rational matrices with binary encoding length, σ, and that we run an SKM method on the normalized system $\tilde{A}x \leq \tilde{b}$ $\left(\text{where } \tilde{a}_i = \frac{1}{||a_i||}a_i \text{ and } \tilde{b}_i = \frac{1}{||a_i||}b_i\right)$ with $x_0 = 0$. Suppose the number of iterations k satisfies

$$k > \frac{4\sigma - 4 - \log n + 2\log\left(\max_{j\in[m]}||a_j||\right)}{\log\left(\frac{mL_2^2}{mL_2^2 - 2\lambda + \lambda^2}\right)}.$$

If the system $Ax \leq b$ is feasible, the probability that the iterate x_k is not a certificate of feasibility is at most

$$\frac{\max||a_j||\ 2^{2\sigma-2}}{n^{1/2}}\left(1 - \frac{2\lambda - \lambda^2}{mL_2^2}\right)^{k/2},$$

which decreases with k.

The final contribution of our paper is a small computational study presented in Section 3. The main purpose of our experiments is not to compare the running times versus established methods. Rather, we wanted to determine how our new algorithms compare with the classical algorithms of Agmon, Motzkin and Schoenberg, and Kaczmarz. We examine how the sampling and projection parameters affects the performance of SKM. We try different types of data, but we assume in most of the data that the number of rows m is large, much larger than n. The reason is that this is the regime in which the SKM methods are most relevant and often the only alternative. Iterated-project methods are truly interesting in cases where the number of constraints is very large (possibly so large it is unreadable in memory) or when the constraints can only be sampled due to uncertainty or partial information. Such regimes arise naturally in applications of machine learning [CE14] and in online linear programming (see [AWY14] and its references). Finally, it has already been shown in prior experiments that, for typical small values of m, n where the system can be read entirely, iterated-projection methods are not able to compete with the simplex method (see [BDJ14, HMSW53]). Here we compare our SKM code with MATLAB's interior-point methods and active set methods code. We also compare SKM with another iterated projection method, the block Kaczmarz method [NT13].

2 Proof of Theorem 1

We show that the SKM methods enjoy a linear rate of convergence. We begin with a simple useful observation.

Lemma 2. Suppose $\{a_i\}_{i=1}^n, \{b_i\}_{i=1}^n$ are real sequences so that $a_{i+1} > a_i > 0$ and $b_{i+1} \geq b_i \geq 0$. Then

$$\sum_{i=1}^n a_i b_i \geq \sum_{i=1}^n \bar{a} b_i, \text{ where } \bar{a} \text{ is the average } \bar{a} = \frac{1}{n}\sum_{i=1}^n a_i.$$

Proof. Note that $\sum_{i=1}^{n} a_i b_i = \sum_{i=1}^{n} \bar{a} b_i + \sum_{i=1}^{n} (a_i - \bar{a}) b_i$, so we need only show that $\sum_{i=1}^{n} (a_i - \bar{a}) b_i \geq 0$, which is equivalent to $\sum_{i=1}^{n} (n a_i - \sum_{j=1}^{n} a_j) b_i \geq 0$, so we define the coefficients $c_i := n a_i - \sum_{j=1}^{n} a_j$. Now, since $\{a_i\}_{i=1}^{n}$ is strictly increasing, there is some $1 < k < n$ so that $c_k \leq 0$ and $c_{k+1} > 0$ and the c_i are strictly increasing. Since $\{b_i\}_{i=1}^{n}$ is non-negative and non-decreasing we have

$$\sum_{i=1}^{n} c_i b_i = \sum_{i=1}^{k} c_i b_i + \sum_{i=k+1}^{n} c_i b_i \geq \sum_{i=1}^{k} c_i b_k + \sum_{i=k+1}^{n} c_i b_k = b_k \sum_{i=1}^{n} c_i = 0.$$

Thus, we have $\sum_{i=1}^{n} a_i b_i = \sum_{i=1}^{n} \bar{a} b_i + \sum_{i=1}^{n} (a_i - \bar{a}) b_i \geq \sum_{i=1}^{n} \bar{a} b_i$. □

Proof. (of Theorem 1) Denote by $\mathcal{P}$ the projection operator onto the feasible region P, and write s_j for the number of zero entries in the residual $(Ax_j - b)^+$, which correspond to satisfied constraints. Define $V_j := \max\{m - s_j, m - \beta + 1\}$. Recalling that the method defines $x_{j+1} = x_j - \lambda (A_{\tau_j} x_j - b_{\tau_j})^+_{i^*} a_{i^*}$ where $i^* = \underset{i \in \tau_j}{\operatorname{argmax}} \{a_i^T x_j - b_i, 0\} = \underset{i \in \tau_j}{\operatorname{argmax}} (A_{\tau_j} x_j - b_{\tau_j})^+_i$, we have

$$\begin{aligned} d(x_{j+1}, P)^2 &= \|x_{j+1} - \mathcal{P}(x_{j+1})\|^2 \leq \|x_{j+1} - \mathcal{P}(x_j)\|^2 = \|x_j - \lambda (A_{\tau_j} x_j - b_{\tau_j})^+_{i^*} a_{i^*} - \mathcal{P}(x_j)\|^2 \\ &= \|x_j - \mathcal{P}(x_j)\|^2 + \lambda^2 ((A_{\tau_j} x_j - b_{\tau_j})^+_{i^*})^2 \|a_{i^*}\|^2 - 2\lambda (A_{\tau_j} x_j - b_{\tau_j})^+_{i^*} a_{i^*}^T (x_j - \mathcal{P}(x_j)). \end{aligned}$$

Since $a_{i^*}^T (x_j - \mathcal{P}(x_j)) \geq a_{i^*}^T x_j - b_{i^*}$, we have that

$$\begin{aligned} d(x_{j+1}, P)^2 &\leq d(x_j, P)^2 + \lambda^2 ((A_{\tau_j} x_j - b_{\tau_j})^+_{i^*})^2 \|a_{i^*}\|^2 - 2\lambda (A_{\tau_j} x_j - b_{\tau_j})^+_{i^*} (a_{i^*}^T x_j - b_{i^*}) \\ &= d(x_j, P)^2 - (2\lambda - \lambda^2) ((A_{\tau_j} x_j - b_{\tau_j})^+_{i^*})^2 \\ &= d(x_j, P)^2 - (2\lambda - \lambda^2) \|(A_{\tau_j} x_j - b_{\tau_j})^+\|_\infty^2. \end{aligned} \tag{1}$$

Now, we take advantage of the fact that, if we consider the size of the entries of $(Ax_j - b)^+$, we can determine the precise probability that a particular entry of the residual vector is selected. Let $(Ax_j - b)^+_{i_k}$ denote the $(k + \beta)$th smallest entry of the residual vector (i.e., if we order the entries of $(Ax_j - b)^+$ from smallest to largest, we denote by $(Ax_j - b)^+_{i_k}$ the entry in the $(k + \beta)$th position). Each sample has equal probability of being selected, $\binom{m}{\beta}^{-1}$. However, the frequency that each entry of the residual vector will be expected to be selected (in Step 3 of SKM) depends on its size. The βth smallest entry will be selected from only one sample, while the m-th smallest entry (i.e., the largest entry) will be selected from all samples in which it appears. Each entry is selected according to the number of samples in which it appears and is largest. Thus, if we take expectation of both sides (with respect to the probabilistic choice of sample, τ_j, of size β), then

$$\mathbb{E}[\|(A_{\tau_j} x_j - b_{\tau_j})^+\|_\infty^2] = \frac{1}{\binom{m}{\beta}} \sum_{k=0}^{m-\beta} \binom{\beta - 1 + k}{\beta - 1} ((Ax_j - b)^+_{i_k})^2 \tag{2}$$

$$\geq \frac{1}{\binom{m}{\beta}} \sum_{k=0}^{m-\beta} \frac{\sum_{\ell=0}^{m-\beta} \binom{\beta-1+\ell}{\beta-1}}{m - \beta + 1} ((Ax_j - b)^+_{i_k})^2 \tag{3}$$

$$= \sum_{k=0}^{m-\beta} \frac{1}{m - \beta + 1} ((Ax_j - b)^+_{i_k})^2 \tag{4}$$

$$\geq \frac{1}{m - \beta + 1} \min\left\{ \frac{m - \beta + 1}{m - s_j}, 1 \right\} \|(Ax_j - b)^+\|_2^2, \tag{5}$$

where (3) follows from Lemma 2, because $\{\binom{\beta-1+k}{\beta-1}\}_{k=0}^{m-\beta}$ is strictly increasing and $\{(Ax_j-b)^+_{i_k}\}_{k=0}^{m-\beta}$ is non-decreasing. Equality (4) follows from (3) due to the fact that $\sum_{\ell=0}^{m-\beta}\binom{\beta-1+\ell}{\beta-1} = \binom{m}{\beta}$ which is known as the column-sum property of Pascal's triangle, among other names. Inequality (5) follows from the fact that the ordered summation in (4) is at least $\frac{m-\beta+1}{m-s_j}$ of the norm of the residual vector (since s_j of the entries are zero) or is the entire residual vector provided $s_j \geq \beta - 1$.

Thus, we have

$$\begin{aligned}\mathbb{E}[d(x_{j+1},P)^2] &\leq d(x_j,P)^2 - (2\lambda-\lambda^2)\mathbb{E}[\|(A_{\tau_j}x_j - b_{\tau_j})^+\|_\infty^2] \\ &\leq d(x_j,P)^2 - \frac{2\lambda-\lambda^2}{V_j}\|(Ax_j-b)^+\|_2^2 \leq \left(1-\frac{2\lambda-\lambda^2}{V_jL_2^2}\right)d(x_j,P)^2.\end{aligned}$$

Since $V_j \leq m$ in each iteration,

$$\mathbb{E}[d(x_{j+1},P)^2] \leq \left(1-\frac{2\lambda-\lambda^2}{mL_2^2}\right)d(x_j,P)^2.$$

Thus, inductively, we get that

$$\mathbb{E}[d(x_k,P)^2] \leq \left(1-\frac{2\lambda-\lambda^2}{mL_2^2}\right)^k d(x_0,P)^2.$$

□

Now, we have that the SKM methods will perform at least as well as the Randomized Kaczmarz method in expectation; however, if we know that after a certain point the iterates satisfy some of the constraints, we can improve our expected convergence rate guarantee. Clearly, after the first iteration, if $\lambda \geq 1$, in every iteration at least one of the constraints will be satisfied so we can guarantee a very slightly increased expected convergence rate. However, we can say more based on the geometry of the problem.

Lemma 3. The sequence of iterates, $\{x_k\}$ generated by an SKM method are pointwise closer to the feasible polyhedron P. That is, for all $a \in P$, $\|x_k - a\| \leq \|x_{k-1} - a\|$ for all iterations k.

Proof. For $a \in P$, $\|x_k - a\| \leq \|x_{k-1} - a\|$ for all k since $a \in P \subset H_{t_k} := \{x : a_{t_k}^T x \leq b_{t_k}\}$ and x_k is the projection of x_{k-1} towards or into the half-space H_{t_k} (provided $x_{k-1} \notin H_{t_k}$, in which case the inequality is true with equality). □

Lemma 4. If P is n-dimensional (full-dimensional) then the sequence of iterates $\{x_k\}$ generated by an SKM method converge to a point $l \in P$.

Proof. Let $a \in P$. Note that the limit, $\lim_{k\to\infty}\|x_k - a\| =: r_a$ exists since $\{\|x_k - a\|\}$ is bounded and decreasing (with probability 1). Define

$$S(a) := \{x : \|x-a\| = r_a\} \text{ and } X := \bigcap_{a\in P} S(a).$$

Note that X is not empty since the bounded sequence $\{x_k\}$ must have a limit point, l, achieving $\|l-a\| = r_a$. Moreover, suppose there were two such points, $l, l' \in X$. Define $\pi := \{x : \|l-x\| = \|l'-x\|\}$ to be the

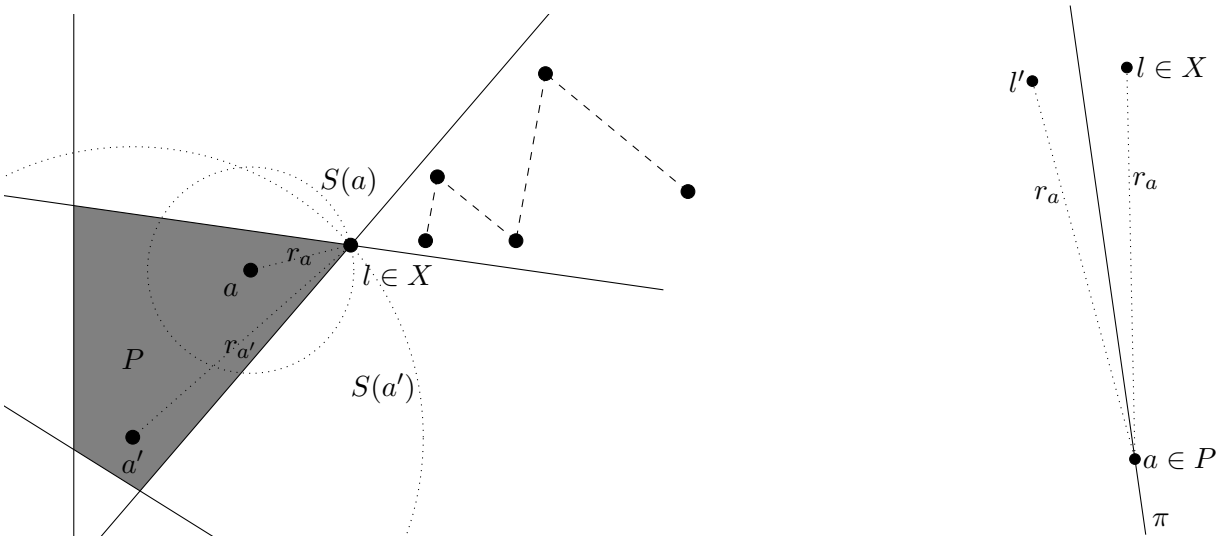

Figure 3: Left: image of $a \in P$, r_a and $S(a)$ and $l \in \bigcap_{a \in P} S(a)$ as defined in Lemma 4. Right: image of $l, l' \in X$ contradicting the full-dimensionality of P.

hyperplane of points equidistance between l, l'. Then for $a \in P$, we have $l, l' \in S(a)$. Hence, $a \in \pi$ and we have that $P \subset \pi$, which contradicts the full dimensionality of P. Thus X contains only one point, l, and it must be a limit point of $\{x_k\}$. Now, since $\{x_k\}$ is converging to P (with probability one), we must have that $l \in P$.

Now, suppose that $x_k \not\to l$ (i.e. only a subsequence of $\{x_k\}$ converges to l). Thus, there exists an $\epsilon > 0$ so that for all K there exists $k \geq K$ with $\|x_k - l\| > \epsilon$. However, there exists a subsequence of $\{x_k\}$ which is converging to l, so there must exist some K_1 with $\|x_{K_1} - l\| < \epsilon$. Thus, at some point the sequence $\|x_k - l\|$ must increase, which contradicts Lemma 3. Hence, $x_k \to l$. □

Lemma 5. Let l be the limit point of the $\{x_k\}$. There exists an index K so that if $a_j^T l < b_j$ then $a_j^T x_k \leq b_j$ for all $k \geq K$.

Proof. This is obvious from $x_k \to l$. □

We would like to conclude with a small "qualitative" proposition that indicates there are two stages of behavior of the SKM algorithms. After the K-th iteration the point is converging to a particular face of the polyhedron. At that moment one has essentially reduced the calculation to an equality system problem, because the inequalities that define the face of convergence need to be met with equality in order to reach the polyhedron.

Proposition 3. If the feasible region P is generic and nonempty (i.e., full-dimensional and every vertex satisfies exactly n constraints with equality), then an SKM method with samples of size $\beta \leq m - n$ will converge to a single face F of P and all but the constraints defining F will eventually be satisfied. Thus, the method is guaranteed an increased convergence rate after some index K; for $k \geq K$

$$\mathbb{E}[d(x_k, P)^2] \leq \left(1 - \frac{2\lambda - \lambda^2}{mL_2^2}\right)^K \left(1 - \frac{2\lambda - \lambda^2}{(m - \beta + 1)L_2^2}\right)^{k-K} d(x_0, P)^2.$$

Proof. (of Proposition 3)) Since a generic polyhedron is full-dimensional, by Lemma 4, we have that the SKM method iterates converge to a point on the boundary of P, l. Now, since this l lies on a face of P and P is generic, this face is defined by at most n constraints. By Lemma 5, there exists K so that for $k \geq K$ at least $m - n$ of the constraints have been satisfied. Thus, our proposition follows from Theorem 1. □

2.1 Proof of Theorem 2

Now, we show that the general SKM method (when $\lambda \neq 2$) on rational data is finite in expectation.

We will additionally make use of the following lemma (which is key in demonstrating that Khachian's ellipsoidal algorithm is finite and polynomial-time [Hač79]) in our proof:

Lemma 6. If the rational system $Ax \leq b$ is feasible, then there is a feasible solution $\hat{x}$ whose coordinates satisfy $|\hat{x}_j| \leq \frac{2^\sigma}{2n}$ for $j = 1, ..., n$.

Using the bound on the expected distance to the solution polyhedron, P, we can show a bound on the expected number of iterations needed to detect feasibility (which does not depend on the size of block selected).

Proof. (of Theorem 2) First, note that if $\tilde{P} := \{x | \tilde{A}x \leq \tilde{b}\}$, then $P = \tilde{P}$. Then, by Lemma 6, if $\tilde{A}x \leq \tilde{b}$ is feasible (so $Ax \leq b$ is feasible) then there is a feasible solution $\hat{x}$ with $|\hat{x}_j| < \frac{2^\sigma}{2n}$ for all $j = 1, 2, ..., n$ (here σ is the binary encoding length for the unnormalized A, b). Thus, since $x_0 = 0$,

$$d(x_0, P) = d(x_0, \tilde{P}) \leq ||\hat{x}|| \leq \frac{2^{\sigma-1}}{n^{1/2}}.$$

Now, define $\tilde{\theta}(x)$ to be the maximum violation for the new, normalized system $\tilde{A}x \leq \tilde{b}$,

$$\tilde{\theta}(x) := \max\{0, \max_{i\in[m]} \tilde{a}_i^T x - \tilde{b}_i\} = \max\left\{0, \max_{i\in[m]} \frac{a_i^T x - b_i}{||a_i||}\right\}.$$

By Lemma 1, if the system $\tilde{A}x \leq \tilde{b}$ is infeasible (so $Ax \leq b$ is infeasible), then

$$\tilde{\theta}(x) = \max\{0, \max_{i\in[m]} \frac{a_i^T x - b_i}{||a_i||}\} \geq \frac{\max\{0, \max_{i\in[m]} a_i^T x - b_i\}}{\max_{j\in[m]} ||a_j||} = \frac{\theta(x)}{\max_{j\in[m]} ||a_j||} \geq \frac{2^{1-\sigma}}{\max_{j\in[m]} ||a_j||}.$$

When running SKM on $\tilde{A}x \leq \tilde{b}$, we can conclude that the system is feasible when $\tilde{\theta}(x) < \frac{2^{1-\sigma}}{\max_{j\in[m]} ||a_j||}$.

Now, since every point of P is inside the half-space defined by $\{x | \tilde{a}_i^T x \leq \tilde{b}_i\}$ for all $i = 1, \cdots, m$, we have $\tilde{\theta}(x) = \max\{0, \max_{i\in[m]} \tilde{a}_i^T x - \tilde{b}_i\} \leq d(x, P)$. Therefore, if $Ax \leq b$ is feasible, then

$$\mathbb{E}(\tilde{\theta}(x_k)) \leq \mathbb{E}(d(x_k, P)) \leq \left(1 - \frac{2\lambda - \lambda^2}{mL_2^2}\right)^{k/2} d(x_0, P) \leq \left(1 - \frac{2\lambda - \lambda^2}{mL_2^2}\right)^{k/2} \frac{2^{\sigma-1}}{n^{1/2}},$$

where the second inequality follows from Theorem 1 and the third inequality follows from Lemma 6 and the discussion above.

Now, we anticipate to have detected feasibility when $\mathbb{E}(\tilde{\theta}(x_k)) < \frac{2^{1-\sigma}}{\max\limits_{j\in[m]} ||a_j||}$, which is true for

$$k > \frac{4\sigma - 4 - \log n + 2\log\left(\max\limits_{j\in[m]}||a_j||\right)}{\log\left(\frac{mL_2^2}{mL_2^2 - 2\lambda + \lambda^2}\right)}.$$

Furthermore, by Markov's inequality (see e.g., [She02, Section 8.2]), if the system $Ax \leq b$ is feasible, then the probability of not having a certificate of feasibility is bounded:

$$\mathbb{P}\left(\tilde{\theta}(x_k) \geq \frac{2^{1-\sigma}}{\max\limits_{j\in[m]} ||a_j||}\right) \leq \frac{\mathbb{E}(\tilde{\theta}(x_k))}{\frac{2^{1-\sigma}}{\max\limits_{j\in[m]} ||a_j||}} < \frac{\left(1 - \frac{2\lambda-\lambda^2}{mL_2^2}\right)^{k/2} \frac{2^{\sigma-1}}{n^{1/2}}}{\frac{2^{1-\sigma}}{\max\limits_{j\in[m]} ||a_j||}} = \frac{2^{2\sigma-2}\max\ ||a_j||}{n^{1/2}}\left(1 - \frac{2\lambda-\lambda^2}{mL_2^2}\right)^{k/2}.$$

This completes the proof. □

3 Experiments

We implemented the SKM methods in MATLAB [MAT16] on a 32GB RAM 8-node cluster (although we did not exploit any parallelization), each with 12 cores of Intel Xeon E5-2640 v2 CPUs running at 2 GHz, and ran them on systems while varying the projection parameter, λ, and the sample size, β. We divided our tests into three broad categories: random data, non-random data, and comparisons to other methods. Our experiments focus on the regime $m \gg n$, since as mentioned earlier, this is the setting in which iterative methods are usually applied; however, we see similar behavior in the underdetermined setting as well.

3.1 Experiments on random data

First we considered systems $Ax \leq b$ where A has entries consisting of standard normal random variables and b is chosen to force the system to have a solution set with non-empty interior (we generated a consistent system of equations and then perturbed the right hand side with the absolute value of a standard normal error vector). We additionally considered systems where the rows of A are highly correlated (each row consists only of entries chosen uniformly at random from $[.9, 1]$ or only of entries chosen uniformly at random from $[-1, -.9]$) and b is chosen as above. We vary the size of $A \in \mathbb{R}^{m\times n}$, which we note in each example presented below.

In Figure 3.1, we provide experimental evidence that for each problem there is an optimal choice for the sample size, β, in terms of computation. We measure the average computational time necessary for SKM with several choices of sample size β to reach halting (positive) residual error 2^{-14} (i.e. $||(Ax_k - b)^+||_2 \leq 2^{-14}$). Regardless of choice of projection parameter, λ, we see a minimum for performance occurs for β between 1 and m.

For the experiments in Figures 3.1, 3.1, and 3.1, we fixed the projection parameter at $\lambda = 1.6$ (for reasons discussed below). On the left of Figure 3.1, we see the residual error decreases more quickly per iteration as the sample size, β increases. However, on the right, when measuring the computational time, SKM with $\beta \approx 5000$ performs best.

In Figure 3.1, we ran experiments varying the halting error and see that the sample size selection, β, depends additionally on the desired final distance to the feasible region, P. On the right, we attempted to pinpoint the optimal choice of β by reducing the sample sizes we were considering.

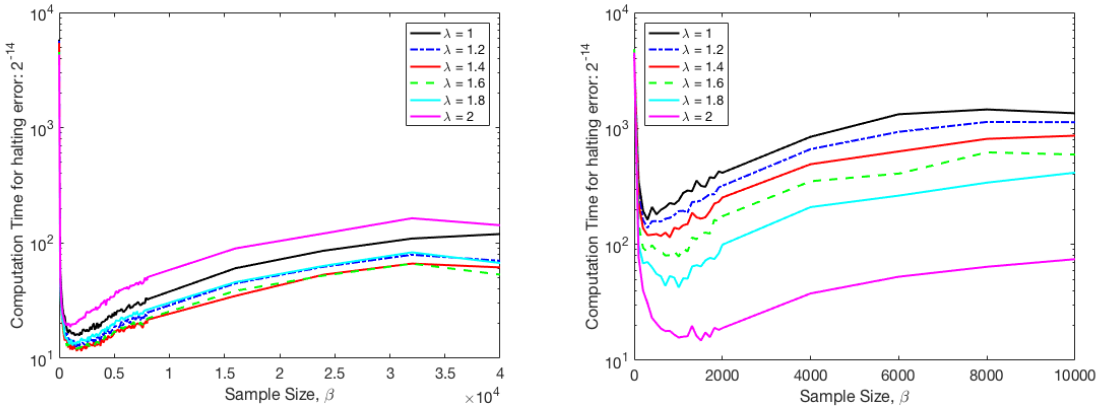

Figure 4: Left: Average comp. time for SKM on 40000×100 Gaussian system to reach residual error 2^{-14}. Right: Average comp. time for SKM on 10000×100 correlated random system to reach residual error.

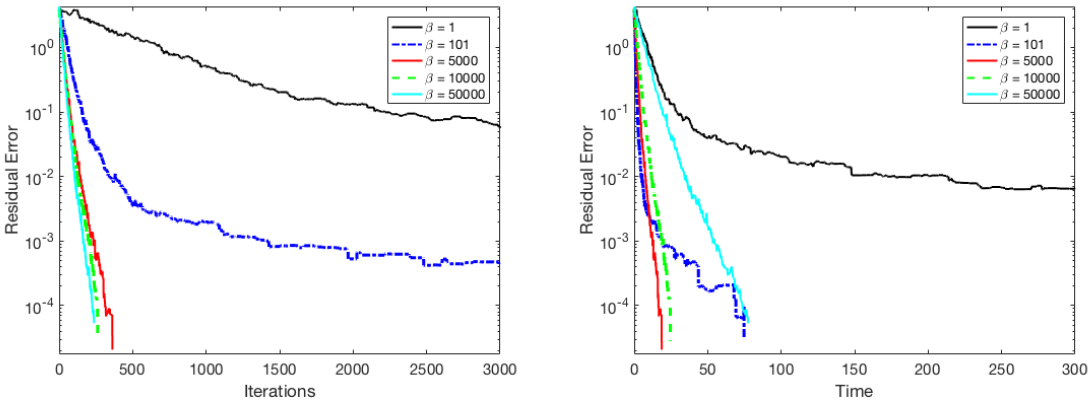

Figure 5: Left: Iterations vs. residual error for SKM with various sample sizes on 50000×100 Gaussian system. Right: Time vs. residual error.

Like [SV09], we observe that 'overshooting' ($\lambda > 1$) outperforms other projection parameters, $\lambda \leq 1$. In Figure 3.1, we see that the optimal projection parameter, λ is system dependent. For the experiments in Figure 3.1, we ran SKM on the same system until the iterates had residual error less than 2^{-14} and averaged the computational time taken over ten runs. The best choice of λ differed greatly between the Gaussian random systems and the correlated random systems; for Gaussian systems it was $1.4 < \lambda < 1.6$ while for correlated systems it was $\lambda = 2$.

Our bound on the distance remaining to the feasible region decreases as the number of satisfied constraints increases. In Figure 3.1, we see that the fraction of satisfied constraints initially increased most quickly for SKM with sample size, $1 < \beta < m$ and projection parameter, $\lambda > 1$. On the left, we show that SKM with $\beta = m$ is faster in terms of number of iterations. However, on the right, we show that SKM with $1 < \beta < m$ outperforms $\beta = m$ in terms of time because of its computational cost in each iteration.

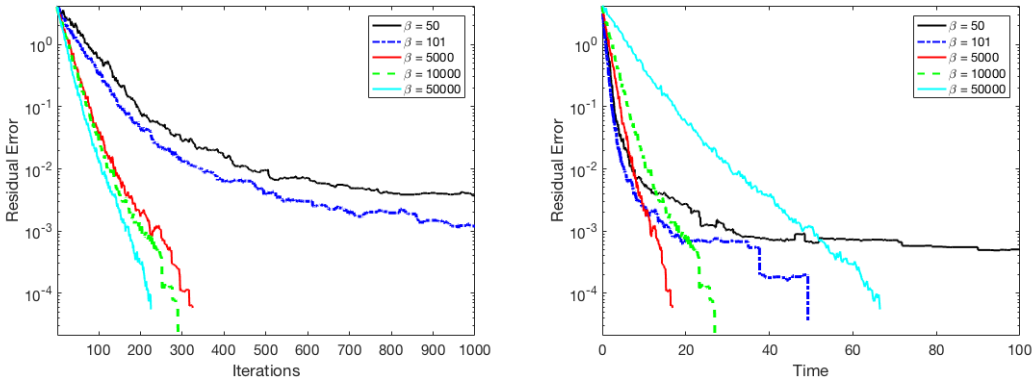

Figure 6: Left: Iterations vs. residual error for SKM with sample sizes from 50 to m on 50000×100 Gaussian system. Right: Time vs. residual error.

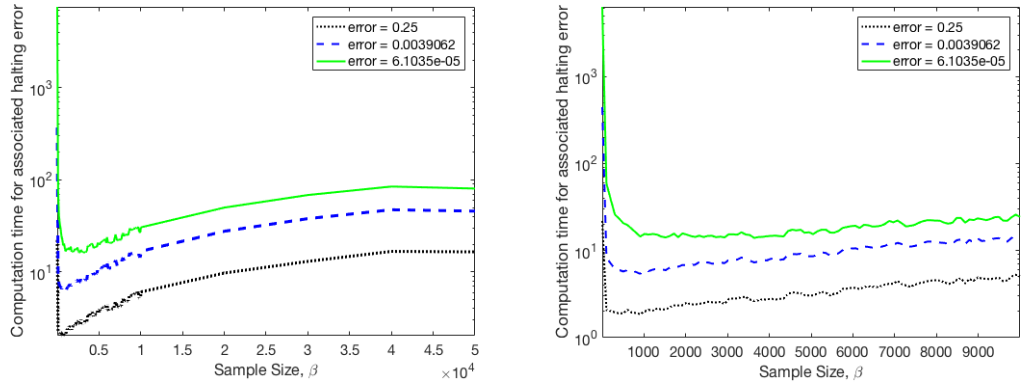

Figure 7: Left: Average comp. time for SKM on 50000×100 Gaussian system to reach various residual errors for β between 1 and m. Right: Average comp. time for β between 1 and $m/5$.

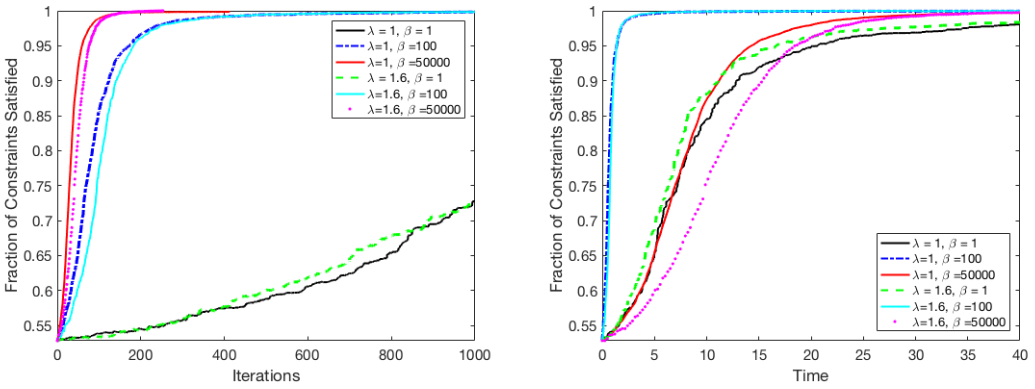

Figure 8: Left: Iterations vs. fraction of contraints satisfied for SKM methods on 50000×100 Gaussian system. Right: Time vs. fraction of contraints satisfied.

3.2 Experiments on non-random data

We consider next some non-random, non-fabricated test problems: support vector machine (SVM) linear classification instances and feasibility problems equivalent to linear programs arising in well-known benchmark libraries.

We first consider instances that fit the classical SVM problem (see [CE14]). We used the SKM methods to solve the SVM problem (find a linear classifier) for several data sets from the UCI Machine Learning Repository [Lic13]. The first data set is the well-known Wisconsin (Diagnostic) Breast Cancer data set, which includes data points (vectors) whose features (components) are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image. Each data point is classified as malignant or benign. The resulting solution to the homogenous system of inequalities, $Ax \leq 0$ would ideally define a hyperplane which separates given malignant and benign data points. However, this data set is not separable. The system of inequalities has $m = 569$ constraints (569 data points) and $n = 30$ variables (29 data features). Here, SKM is minimizing the residual norm, $||Ax_k||_2$ and is run until $||Ax_k||_2 \leq 0.5$. See Figure 9 for results of SKM runtime on this data set.

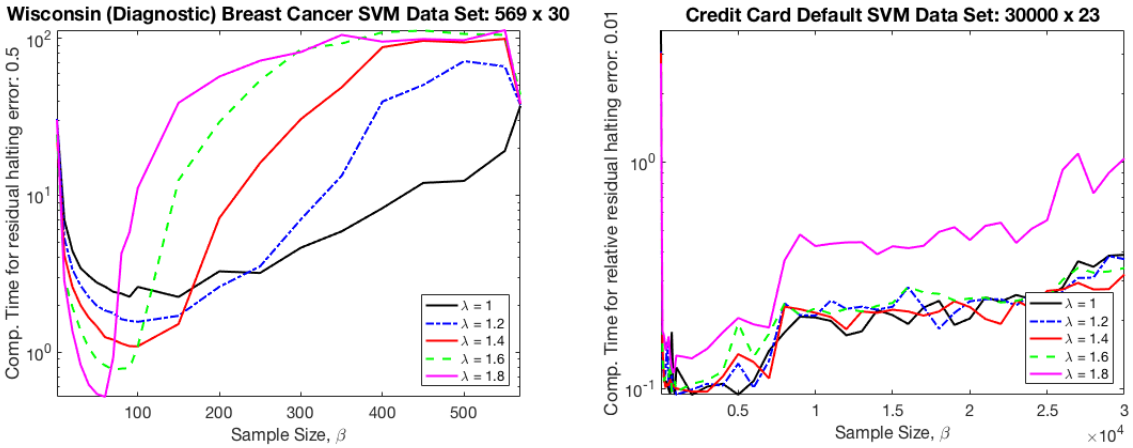

Figure 9: Left: Breast Cancer Data SVM. Right: Credit Card Data SVM.

The second data set is a credit card data set, whose data points include features describing the payment profile of a credit card user and the binary classification is for on-time payment or default payment in a billing cycle [YL09]. The resulting solution to the homogenous system of inequalities would ideally define a hyperplane which separates given on-time and default data points. However, this data set is not separable. The system of inequalities has $m = 30000$ (30000 credit card user profiles) and $n = 23$ (22 profile features). Here, SKM is run until $||Ax_k||_2/||Ax_0||_2 \leq 0.01$. See Figure 9 for results of SKM runtime on this data set.

In the experiments, we again see that for each problem there is an optimal choice for the sample size, β, in terms of smallest computation time. We measure the average computation time necessary for SKM with several choices of sample size β to reach the halting (positive) residual error. Regardless of choice of projection parameter, λ, we see again that best performance occurs for β between 1 and m. Note that the curves are not as smooth as before, which we attribute to the wider irregularity of coefficients, which in turn forces the residual error more to be more dependent on the actual constraints.

We next implemented SKM on several *Netlib* linear programming (LP) problems [Net]. Each of these problems was originally formulated as the LP $\min c^T x$ subject to $Ax = b,\ l \leq x \leq u$ with optimum value p^*. We reformulated these problems as the equivalent linear feasibility problem $\tilde{A}x \leq \tilde{b}$ where

$$\tilde{A} = \begin{bmatrix} A \\ -A \\ I \\ -I \\ c^T \end{bmatrix} \text{ and } \tilde{b} = \begin{bmatrix} b \\ -b \\ u \\ -l \\ p^* \end{bmatrix}.$$

See Figures 10, 11, 12, 13, and 14 for results of SKM runtime on these problems as we vary β and λ. Once more, regardless of choice of projection parameter, λ, we see optimal performance occurs for β between 1 and m.

It would be possible to handle these equalities without employing our splitting technique to generate inequalities. This splitting technique only increases m ($||A||_F^2$) and does not affect the Hoffman constant, which is $||\tilde{A}^{-1}||_2$ in this case. It may be useful to explore such an extension.

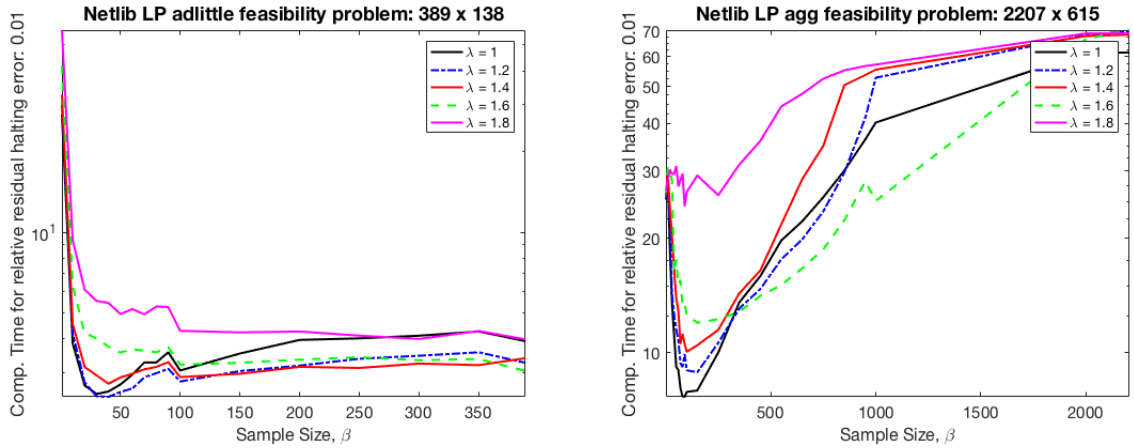

Figure 10: Left: SKM behavior for *Netlib* LP adlittle. Right: SKM behavior for *Netlib* LP agg

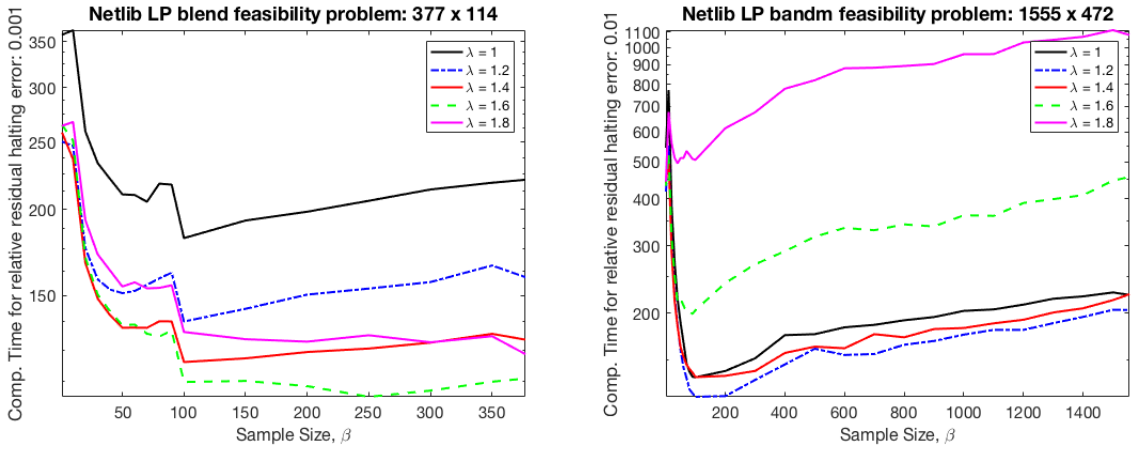

Figure 11: Left: SKM behavior for *Netlib* LP blend. Right: SKM behavior for *Netlib* LP bandm.

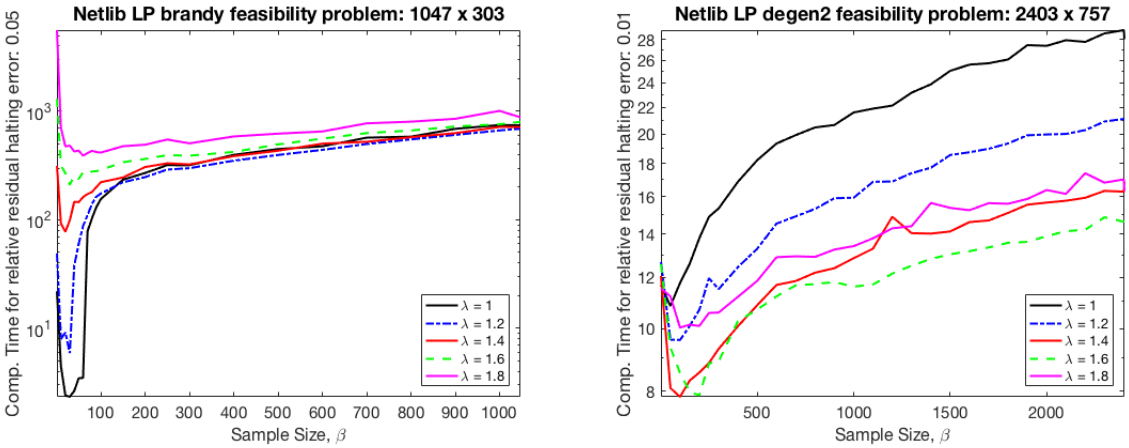

Figure 12: Left: SKM behavior for *Netlib* LP brandy. Right: SKM behavior for *Netlib* LP degen2.

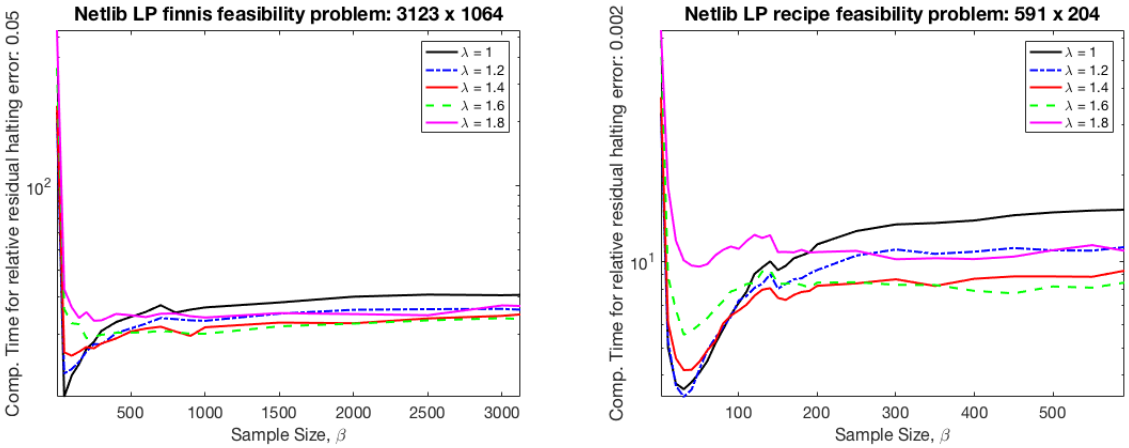

Figure 13: Left: SKM behavior for *Netlib* LP finnis. Right: SKM behavior for *Netlib* LP recipe.

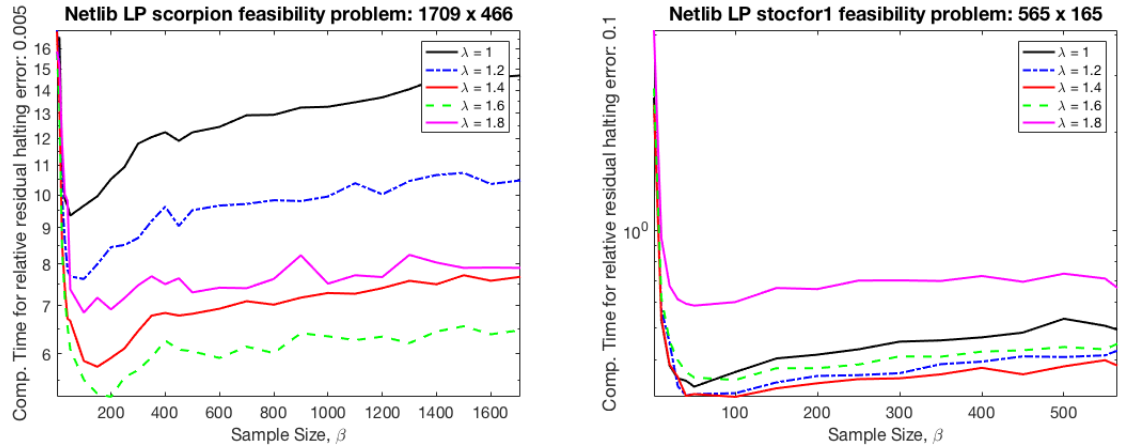

Figure 14: Left: SKM behavior for *Netlib* LP scorpion. Right: SKM behavior for *Netlib* LP stocfor1.

3.3 Comparison to existing methods

In Table 1, we investigate the performance behavior of SKM versus interior-point and active-set methods on several *Netlib* LPs. For fairness of comparison, we gauge our code written in MATLAB versus the MATLAB Optimization Toolbox function *fmincon*. The function *fmincon* allows a user to select either an 'interior-point' algorithm or an 'active-set' algorithm.

We first used fmincon to solve the feasibility problem as described in Section 3.2 by applying this function to $\min 0$ such that $\tilde{A}x \leq \tilde{b}$. However, the interior-point method and active-set method were mostly unable to solve these feasibility form problems. The interior-point algorithm was never able to solve feasibility, due to the fact that the system of equations defined by the KKT conditions in each iteration was numerically singular. Similarly, in most cases, the active-set method was halted in the initial step of finding a feasible point. For fairness of comparison, we do not list these results.

In Table 1, we list CPU timings for the MATLAB interior-point and active-set fmincon algorithms to solve the original optimization LPs ($\min c^T x$ such that $Ax = b, l \leq x \leq u$), and SKM to solve the equivalent feasibility problem, $\tilde{A}x \leq \tilde{b}$, as described in Section 3.2. Note that this is not an obvious comparion as SKM is designed for feasibility problems, and in principle, the stopping criterion may force SKM to stop near a feasible point, but not necessarily near an optimum. On the other hand, interior point methods and active set methods decrease the value of the objective and simultaneously solve feasibility. The halting criterion for SKM remains that $\frac{\max(\tilde{A}x_k - \tilde{b})}{\max(\tilde{A}x_0 - \tilde{b})} \leq \epsilon_{\text{err}}$ where ϵ_{err} is the halting error bound listed for each problem in the table. The halting criterion for the fmincon algorithms is that $\frac{\max(Ax_k - b, l - x_k, x_k - u)}{\max(Ax_0 - b, l - x_0, x_0 - u)} \leq \epsilon_{\text{err}}$ and $\frac{c^T x_k}{c^T x_0} \leq \epsilon_{\text{err}}$ where ϵ_{err} is the halting error bound listed for each problem in the table. Each of the methods were started with the same initial point far from the feasible region. The experiments show our SKM method compares favorably with the other codes.

Problem Title	Dimensions	Interior-Point	SKM	Active-Set	ϵ_{err}	SKM λ	SKM β
LP adlittle	389×138	2.08	0.29	1.85	10^{-2}	1.2	30
LP agg	2207×615	109.54*	20.55	554.52*	10^{-2}	1	100
LP bandm	1555×472	27.21	756.71	518.44*	10^{-2}	1.2	100
LP blend	337×114	1.87	367.33	2.20	10^{-3}	1.6	250
LP brandy	1047×303	21.26	240.83	90.46	0.05	1	20
LP degen2	2403×757	6.70	22.41	25725.23	10^{-2}	1.4	100
LP finnis	3123×1064	115.47*	13.76	431380.82*	0.05	1	50
LP recipe	591×204	2.81	2.62	5.56	0.002	1.2	30
LP scorpion	1709×466	11.80	22.22	10.38	0.005	1.6	200
LP stocfor1	565×165	0.53	0.34	3.29	0.1	1.4	50

Table 1: CPU time comparisons for MATLAB methods solving LP and SKM solving feasibility.
$*$ indicates that the solver did not solve the problem to the desired accuracy due to reaching an upper limit on function evaluations of 100000

For the experiments in Table 1, the interior-point method was not able to solve for LP agg and LP finnis before hitting the upper bound on function evaluations due to slow progression towards feasibility. The active-set method was not able to solve for LP agg, LP bandm and LP finnis before hitting the upper bound on function evaluations due to a very slow (or incomplete) initial step in finding a feasible point. As mentioned before, the methods were initialized with a point far from the feasible region which may have contributed to the interior-point and active-set methods poor performances.

In Figures 15 and 16, we compare the SKM method to the block Kaczmarz (BK) method (with randomly selected blocks). Here we solve only systems of linear equations, not inequalities, and we consider

only random data as our implemented block Kaczmarz method selects blocks at random. We see that the performance of the block Kaczmarz method is closely linked to the conditioning of the selected blocks, as the BK method must solve a system of equations in each iteration, rather than one equation as for SKM.

For the Gaussian random data, the selected blocks are well-conditioned and with high probability, the block division has formed a row-paving of the matrix. Here we see that BK outperforms SKM. However, when we consider correlated data instead, the behavior of BK reflects the poor conditioning of the blocks. In the three included figures, we test with correlated matrices with increasingly poorly conditioned blocks. If the blocks are numerically ill-conditioned, SKM is able to outperform BK. For systems of equations in which blocks are well conditioned and easy to identify, BK has advantages over SKM. However, if you are unable or unwilling to find a good paving, SKM can be used and is able to outperform BK. When BK is used with inequalities, a paving with more strict geometric properties must be found, and this can be computationally challenging, see [BN] for details. SKM avoids this issue.

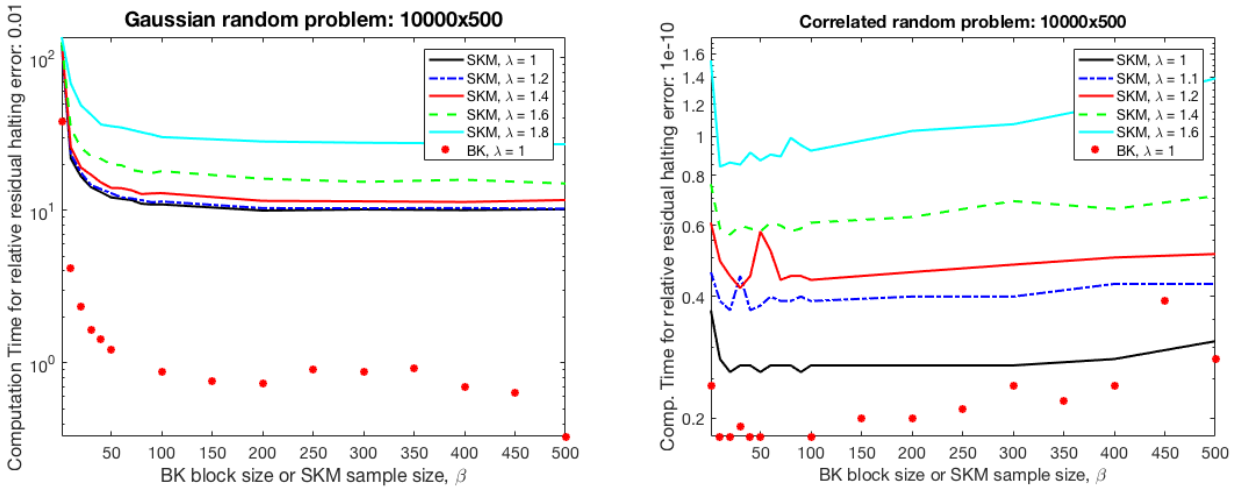

Figure 15: Comparison of SKM method runtimes with various choices of sample size, β and block Kaczmarz method runtimes with various choices of block size on different types of random systems. Left: Gaussian random system. Right: Correlated random system with entries chosen uniformly from $[0.9, 0.9+10^{-5}]$.

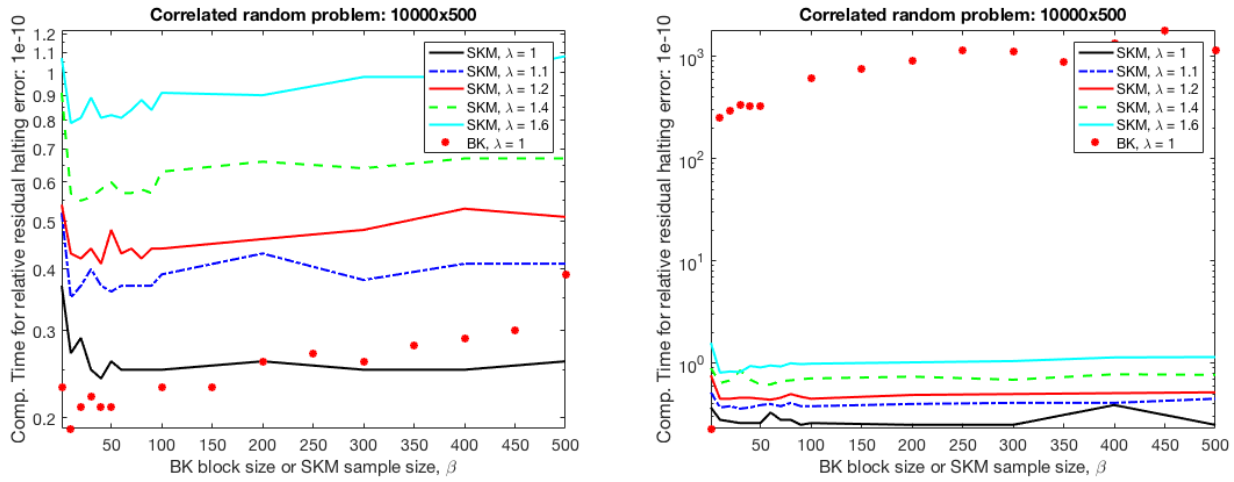

Figure 16: Left: Correlated random system with entries chosen uniformly from $[0.9, 0.9 + 10^{-16}]$. Right: Correlated random system with entries chosen uniformly from $[0.9, 0.9 + 10^{-20}]$.

4 Remarks about optimal selection of parameters

4.1 Choice of β

As observed by Theorem 1, the sample size β used in each iteration of SKM plays a role in the convergence rate of the method. By the definition of V_{k-1} in Theorem 1 and by the bound in Proposition 3 the choice $\beta = m$ yields the fastest convergence rate. Indeed, this coincides with the classical method of Motzkin; one selects the most violated constraint out of *all* the constraints in each iteration. However, it is also clear that this choice of β is extremely costly in terms of computation, and so the more relevant question is about the choice of β that optimizes the convergence rate in terms of total computation.

To gain an understanding of the tradeoff between convergence rate and computation time in terms of the parameter β, we consider a fixed iteration j and for simplicity choose $\lambda = 1$. Denote the residual by $r := (Ax_j - b)^+$, and suppose s inequalities are satisfied in this iteration; that is, r has s zero entries. Write r_{τ_j} for the portion of the residual selected in Step 3 of SKM (so $|\tau_j| = \beta$). Then as seen from Equation (1) in the proof of Theorem 1, the expected improvement (i.e. $d(x_j, P) - d(x_{j+1}, P)$) made in this iteration is given by $\mathbb{E}\|r_{\tau_j}\|_\infty^2$. Expressing this quantity as in (2) along with Lemma 2, one sees that the worst case improvement will be made when the $m - s$ non-zero components of the residual vector are all the same magnitude (i.e. $\mathbb{E}\|r_{\tau_j}\|_\infty \geq \frac{1}{m-s}\|r\|_1$). We thus focus on this scenario in tuning β to obtain a minimax heuristic for the optimal selection. We model the computation count in a fixed iteration as some constant computation time for overhead C plus a factor that scales like $n\beta$, since checking the feasibility of β constraints takes time $O(n\beta)$. We therefore seek a value for β that maximizes the ratio of improvement made and computation cost:

$$\text{gain}(\beta) := \frac{\mathbb{E}\|r_{\tau_j}\|_\infty^2}{C + cn\beta}, \tag{6}$$

when the residual r consists of $m - s$ non-zeros of the same magnitude. Call the support of the residual $T := \text{supp}(r) = \{i : r_i \neq 0\}$. Without loss of generality, we may assume that the magnitude of these entries is just 1. In that case, one easily computes that

$$\mathbb{E}\|r_{\tau_j}\|_\infty^2 = \mathbb{P}(T \cap \tau_j \neq \emptyset) = \begin{cases} 1 - \dfrac{\binom{s}{\beta}}{\binom{m}{\beta}} \approx 1 - \left(\frac{s}{m}\right)^\beta & \text{if } \beta \leq s, \\ 1 & \text{if } \beta > s, \end{cases}$$

where we have used Stirling's approximation in the first case.

We may now plot the quantity

$$\text{gain}(\beta) \approx \frac{1 - \left(\frac{s}{m}\right)^\beta}{C + cn\beta} \tag{7}$$

as a function of β, for various choices of s. Figure 17 shows an example of this function for some specific parameter settings. We see that, as in the experiments of Section 3, optimal β selection need not necessarily be at either of the endpoints $\beta = 1$ or $\beta = m$ (corresponding to classical randomized Kaczmarz and Motzkin's method, respectively). In particular, one observes that as the number of satisfied constraints s increases, the optimal size of β also increases. This of course is not surprising, since with many satisfied constraints if we use a small value of β we are likely to see mostly satisfied constraints in our selection and thus make little to no progress in that iteration. Again, this plot is for the worst case scenario when the residual has constant non-zero entries, but serves as a heuristic for how one might tune the choice of β. In particular, it might be worthwhile to increase β throughout the iterations.

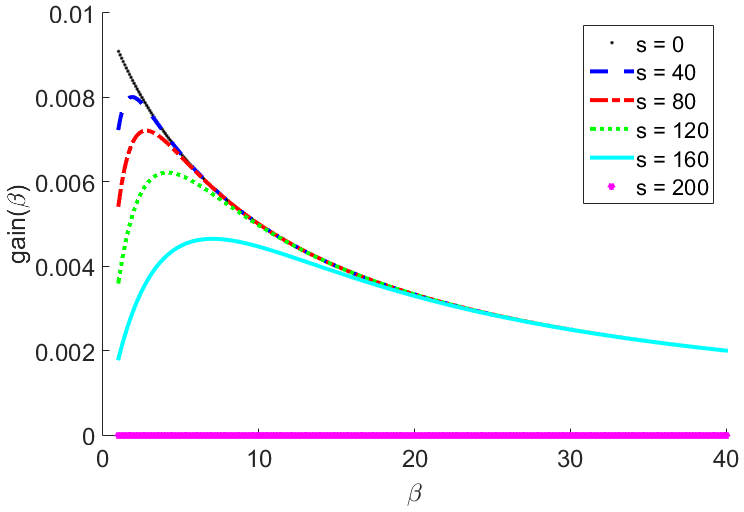

Figure 17: The quantity gain(β) as in (7) as a function of β for various numbers of satisfied constraints s. Here we set $m = 200$, $n = 10$, $c = 1$ and $C = 100$. Optimal values of β maximize the gain function.

4.2 Choice of λ

Additionally, the optimal choice of projection parameter λ is system dependent (e.g., for certain systems, one should choose $\lambda = 1$ while for certain full-dimensional systems, one should choose $\lambda > 1$). Theoretically, the convergence rate we provided in Theorem 1 depends upon λ in a weak way; one would always choose $\lambda = 1$. However, we see experimentally that overshooting outperforms other choices of λ. Additionally, one can easily imagine that for systems whose polyhedral feasible region is full-dimensional, choosing $\lambda > 1$ will outperform $\lambda \leq 1$, as eventually, the iterates could 'hop' into the the feasible region. The proof of Proposition 3 suggests a possible reason why we see this in our experiments. This proposition is a consequence of the fact that if the method does not terminate then it will converge to a unique face of P. If $\lambda > 1$, then this face cannot be a facet of P, as if the method converged to such a face, it would eventually terminate, 'hopping' over the facet into P. Thus, for $\lambda > 1$, the number of possible faces of P that the sequence of iterates can converge to is decreased. Further work is needed before defining the optimal choice of λ or β for any class of systems.

4.3 Concluding remarks

We have shown SKM is a natural generalization of the methods of Kaczmarz and Motzkin with a theoretical analysis that combines earlier arguments. Moreover, compared to these two older methods, the SKM approach leads to significant acceleration with the right choices of parameters. We wish to note that, by easy polarization-homogenization of the information (where the hyperplane normals a_i are thought of as points and the solution vector x is a separating plane), one can reinterpret SKM as a type of *stochastic gradient descent* (SGD). Indeed, in SGD one allows the direction to be a random vector whose expected value is the gradient direction; here we generate a random direction that stems from a sampling of the possible increments. More on this will be discussed in a forthcoming article. In future work we intend to identify the optimal choices for β and λ for classes of systems and to connect SKM to Chubanov's style generation of additional

linear inequalities that have been successfully used to speed computation [Chu12, BDJ14, VZ14]. All code discussed in this paper is freely available at `https://www.math.ucdavis.edu/~jhaddock`.

5 Acknowledgements

The authors are truly grateful to the anonymous referees and the editor for their many comments and suggestions which have greatly improved this paper. The first and second author were partially supported by grant H98230-15-1-0226 from the NSA. The second author was partially supported by a GAANN fellowship. The third author was supported by NSF CAREER #1348721 and the Alfred P. Sloan foundation.

References

[ADG14] H. Avron, A. Druinsky, and A. Gupta. Revisiting asynchronous linear solvers: Provable convergence rate through randomization. In *IEEE 28th Int. Parallel and Distributed Processing Symposium*, pages 198–207. IEEE, 2014.

[Agm54] S. Agmon. The relaxation method for linear inequalities. *Canadian J. Math.*, 6:382–392, 1954.

[AH05] E. Amaldi and R. Hauser. Boundedness theorems for the relaxation method. *Math. Oper. Res.*, 30(4):939–955, 2005.

[AWL14] A. Agaskar, C. Wang, and Y. M. Lu. Randomized Kaczmarz algorithms: Exact MSE analysis and optimal sampling probabilities. In *IEEE Global Conf. on Signal and Information Processing (GlobalSIP)*, pages 389–393. IEEE, 2014.

[AWY14] S. Agrawal, Z. Wang, and Y. Ye. A dynamic near-optimal algorithm for online linear programming. *Operations Research*, 62(4):876–890, 2014.

[BDJ14] A. Basu, J. A. De Loera, and M. Junod. On Chubanov's method for linear programming. *INFORMS Journal on Computing*, 26(2):336–350, 2014.

[Bet04] U. Betke. Relaxation, new combinatorial and polynomial algorithms for the linear feasibility problem. *Discrete Comput. Geom.*, 32(3):317–338, 2004.

[BN] J. Briskman and D. Needell. Block Kaczmarz method with inequalities. *J. Math. Imaging Vis.*, 52(3):385–396.

[CE14] G. Calafiore and L. El Ghaoui. *Optimization Models*. Control systems and optimization series. Cambridge University Press, October 2014.

[CEG83] Y. Censor, P. P. Eggermont, and D. Gordon. Strong underrelaxation in Kaczmarz's method for inconsistent systems. *Numer. Math.*, 41(1):83–92, 1983.

[Cen81] Y. Censor. Row-action methods for huge and sparse systems and their applications. *SIAM Rev.*, 23(4):444–466, 1981.

[Chu12] S. Chubanov. A strongly polynomial algorithm for linear systems having a binary solution. *Math. Programming*, 134(2):533–570, 2012.

[CP12] X. Chen and A. Powell. Almost sure convergence of the Kaczmarz algorithm with random measurements. *J. Fourier Anal. Appl.*, pages 1–20, 2012. 10.1007/s00041-012-9237-2.

[Dum14] B. Dumitrescu. On the relation between the randomized extended Kaczmarz algorithm and coordinate descent. *BIT Numerical Mathematics*, pages 1–11, 2014.

[EHL81] P. P. B. Eggermont, G. T. Herman, and A. Lent. Iterative algorithms for large partitioned linear systems, with applications to image reconstruction. *Linear Algebra Appl.*, 40:37–67, 1981.

[Elf80] T. Elfving. Block-iterative methods for consistent and inconsistent linear equations. *Numer. Math.*, 35(1):1–12, 1980.

[EN11] Y. C. Eldar and D. Needell. Acceleration of randomized Kaczmarz method via the Johnson-Lindenstrauss lemma. *Numer. Algorithms*, 58(2):163–177, 2011. 65F20 (65F10); 2835851; Alexander N. Malyshev.

[GBH70] R. Gordon, R. Bender, and G. T. Herman. Algebraic reconstruction techniques (ART) for three-dimensional electron microscopy and x-ray photography. *J. Theoret. Biol.*, 29:471–481, 1970.

[GO12] M. Griebel and P. Oswald. Greedy and randomized versions of the multiplicative Schwarz method. *Linear Algebra Appl.*, 437(7):1596–1610, 2012.

[Gof80] J.-L. Goffin. The relaxation method for solving systems of linear inequalities. *Math. Oper. Res.*, 5(3):388–414, 1980.

[Gof82] J.-L. Goffin. On the nonpolynomiality of the relaxation method for systems of linear inequalities. *Math. Programming*, 22(1):93–103, 1982.

[GPS16] E. Gallopoulos, B. Philippe, and A. H. Sameh. Preconditioners. In *Parallelism in Matrix Computations*, pages 311–341. Springer, 2016.

[GR15] R. M. Gower and P. Richtárik. Randomized iterative methods for linear systems. *SIAM J. Matrix Anal. A.*, 36(4):1660–1690, 2015.

[Hač79] L. G. Hačijan. A polynomial algorithm in linear programming. *Dokl. Akad. Nauk SSSR*, 244(5):1093–1096, 1979.

[HM93] G. T. Herman and L. B. Meyer. Algebraic reconstruction techniques can be made computationally efficient. *IEEE Trans. Medical Imaging*, 12(3):600–609, 1993.

[HMSW53] A. Hoffman, M. Mannos, D. Sokolowsky, and N. Wiegmann. Computational experience in solving linear programs. *Journal of the Society for Industrial and Applied Mathematics*, 1(1):pp. 17–33, 1953.

[HN90] M. Hanke and W. Niethammer. On the acceleration of Kaczmarz's method for inconsistent linear systems. *Linear Algebra Appl.*, 130:83–98, 1990.

[HNR15] A. Hefny, D. Needell, and A. Ramdas. Rows vs. columns: Randomized Kaczmarz or Gauss-Seidel for ridge regression. 2015. Submitted.

[Hof52] A. J. Hoffman. On approximate solutions of systems of linear inequalities. *J. Research Nat. Bur. Standards*, 49:263–265, 1952.

[HS78] C. Hamaker and D. C. Solmon. The angles between the null spaces of x-rays. *J. Math. Anal. Appl.*, 62(1):1–23, 1978.

[Kac37] S. Kaczmarz. Angenäherte auflösung von systemen linearer gleichungen. *Bull.Internat.Acad.Polon.Sci.Lettres A*, pages 335–357, 1937.

[Lic13] M. Lichman. UCI machine learning repository, 2013.

[LL10] D. Leventhal and A. S. Lewis. Randomized methods for linear constraints: convergence rates and conditioning. *Math. Oper. Res.*, 35(3):641–654, 2010.

[LMY15] Y. Li, K. Mo, and H. Ye. Accelerating random Kaczmarz algorithm based on clustering information. *arXiv preprint arXiv:1511.05362*, 2015.

[LW15] J. Liu and S. Wright. An accelerated randomized Kaczmarz algorithm. *Mathematics of Computation*, 2015.

[LWS14] J. Liu, S. J. Wright, and S. Sridhar. An asynchronous parallel randomized Kaczmarz algorithm. *arXiv preprint arXiv:1401.4780*, 2014.

[MAT16] MATLAB. *version 9.0.0 (R2016a)*. The MathWorks Inc., Natick, Massachusetts, 2016.

[MNR15] A. Ma, D. Needell, and A. Ramdas. Convergence properties of the randomized extended gauss-seidel and Kaczmarz methods. *SIAM J. Matrix Anal. A.*, 2015. To appear.

[MS54] T. S. Motzkin and I. J. Schoenberg. The relaxation method for linear inequalities. *Canadian J. Math.*, 6:393–404, 1954.

[MTA81] J.-F. Maurras, K. Truemper, and M. Akgül. Polynomial algorithms for a class of linear programs. *Math. Programming*, 21(2):121–136, 1981.

[Nat01] F. Natterer. *The mathematics of computerized tomography*, volume 32. Society for Industrial and Applied Mathematics, Philadelphia, PA; SIAM, 2001.

[Nee10] D. Needell. Randomized Kaczmarz solver for noisy linear systems. *BIT*, 50(2):395–403, 2010.

[Net] Netlib. The Netlib Linear Programming Library. `www.netlib.org/lp`.

[NSV+16] J. Nutini, B. Sepehry, A. Virani, I. Laradji, M. Schmidt, and H. Koepke. Convergence Rates for Greedy Kaczmarz Algorithms. *UAI*, 2016.

[NSW14a] D. Needell, N. Srebro, and R. Ward. Stochastic gradient descent and the randomized Kaczmarz algorithm. *Math. Programming Series A*, 2014. To appear.

[NSW14b] D. Needell, N. Srebro, and R. Ward. Stochastic gradient descent, weighted sampling, and the randomized Kaczmarz algorithm. In *Proc. Neural Info. Proc. Systems (NIPS)*, 2014.

[NT13] D. Needell and J. A. Tropp. Paved with good intentions: Analysis of a randomized block Kaczmarz method. *Linear Algebra Appl.*, 2013.

[NW13] D. Needell and R. Ward. Two-subspace projection method for coherent overdetermined linear systems. *J. Fourier Anal. Appl.*, 19(2):256–269, 2013.

[NZZ15] D. Needell, R. Zhao, and A. Zouzias. Randomized block Kaczmarz method with projection for solving least squares. *Linear Algebra Appl.*, 484:322–343, 2015.

[OZ15a] P. Oswald and W. Zhou. Convergence analysis for Kaczmarz-type methods in a Hilbert space framework. *Linear Algebra Appl.*, 478:131–161, 2015.

[OZ15b] P. Oswald and W. Zhou. Random reordering in SOR-type methods. *arXiv preprint arXiv:1510.04727*, 2015.

[PP15] S. Petra and C. Popa. Single projection Kaczmarz extended algorithms. *Numerical Algorithms*, pages 1–16, 2015.

[PPKR12] C. Popa, T. Preclik, H. Köstler, and U. Rüde. On Kaczmarz's projection iteration as a direct solver for linear least squares problems. *Linear Algebra Appl.*, 436(2):389–404, 2012.

[RM12] P. Richtárik and T. M. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Math. Programming*, pages 1–38, 2012.

[Ros58] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Cornell Aeronautical Laboratory, Psychological Review*, 65(6):386–408, 1958.

[She02] R. Sheldon. *A first course in probability*. Pearson Education India, 2002.

[SV09] T. Strohmer and R. Vershynin. A randomized Kaczmarz algorithm with exponential convergence. *J. Fourier Anal. Appl.*, 15:262–278, 2009.

[Tan71] K. Tanabe. Projection method for solving a singular system of linear equations and its applications. *Numer. Math.*, 17(3):203–214, 1971.

[Tel82] J. Telgen. On relaxation methods for systems of linear inequalities. *European J. Oper. Res.*, 9(2):184–189, 1982.

[VZ14] L. A. Végh and G. Zambelli. A polynomial projection-type algorithm for linear programming. *Oper. Res. Lett.*, 42(1):91–96, 2014.

[WAL15] C. Wang, A. Agaskar, and Y. M. Lu. Randomized Kaczmarz algorithm for inconsistent linear systems: An exact MSE analysis. *arXiv preprint arXiv:1502.00190*, 2015.

[WM67] T. M. Whitney and R. K. Meany. Two algorithms related to the method of steepest descent. *SIAM J. Numer. Anal.*, 4(1):109–118, 1967.

[XZ02] J. Xu and L. Zikatanov. The method of alternating projections and the method of subspace corrections in Hilbert space. *J. Amer. Math. Soc.*, 15(3):573–597, 2002.

[YL09] I. C. Yeh and C. H. Lien. The comparison of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Systems with Applications*, 36(2):24732480, 2009.

[ZF12] A. Zouzias and N. M. Freris. Randomized extended Kaczmarz for solving least-squares. *SIAM J. Matrix Anal. A.*, 34(2):773–793, 2012.