2-12-2011

Acceleration of Randomized Kaczmarz Method via the Johnson-Lindenstrauss Lemma

Yonina C. Eldar
Israel Institute of Technology

Deanna Needell
Claremont McKenna College

Recommended Citation
Eldar, Y. C., Needell, D., "Acceleration of Randomized Kaczmarz Method via the Johnson-Lindenstrauss Lemma", Numerical Algorithms, vol. 58, num. 2, pp. 163-177, 2011.

Acceleration of Randomized Kaczmarz Method via the Johnson-Lindenstrauss Lemma

Yonina C. Eldar[1], Deanna Needell[2*]

[1]Department of Statistics, Stanford University, Stanford, CA 94305

[2]Department of Electrical Engineering, Technion - Israel Institute of Technology, Haifa 32000

February 15, 2011

Abstract

The Kaczmarz method is an algorithm for finding the solution to an overdetermined consistent system of linear equations $Ax = b$ by iteratively projecting onto the solution spaces. The randomized version put forth by Strohmer and Vershynin yields provably exponential convergence in expectation, which for highly overdetermined systems even outperforms the conjugate gradient method. In this article we present a modified version of the randomized Kaczmarz method which at each iteration selects the optimal projection from a randomly chosen set, which in most cases significantly improves the convergence rate. We utilize a Johnson-Lindenstrauss dimension reduction technique to keep the runtime on the same order as the original randomized version, adding only extra preprocessing time. We present a series of empirical studies which demonstrate the remarkable acceleration in convergence to the solution using this modified approach.

1 Introduction

The Kaczmarz method [18] is a popular algorithm for solving overdetermined consistent systems of linear equations. Due to its simplicity and speed, it has been used in a variety of applications ranging from tomography to digital signal processing [4, 21, 19]. The method uses a series of alternating projections to iteratively converge to the solution of $Ax = b$, and is therefore computationally feasible even for very large systems. Given an initial guess x_0 and denoting by $a_1, \ldots, a_m$ the rows of the $m \times n$ matrix A, each iteration of the method orthogonally projects the current estimation onto the next hyperplane defined as the solutions to $\langle a_i, x\rangle = b_i$, chosen in a cyclic fashion. The algorithm can be described by the iterations:

$$x_{k+1} = x_k + \frac{b[i] - \langle a_i, x_k\rangle}{\|a_i\|_2^2} a_i,$$

where x_k is the kth iterate, $b[i]$ (here and throughout) denotes the ith coordinate of b, and $i = (k \bmod m) + 1$.

Although this technique has been used in practice for quite some time, theoretical guarantees on convergence were difficult to obtain [8, 12, 13]. It is clear that by design of the algorithm,

*Corresponding author: Deanna Needell. Email: dneedell@stanford.edu

the convergence rate depends on the ordering of the rows in A. Therefore, poorly ordered rows can lead to slower convergence. To overcome this difficulty, the rows of A can be selected in a random fashion. It has been observed that this randomized version of the algorithm improves the convergence rate [19, 14], however only recently have theoretical results been obtained [24, 25, 20].

1.1 Randomized Kaczmarz

In [24, 25], Strohmer and Vershynin propose at each iteration to randomly select a row of A with probability proportional to the Euclidean norm of the row. The randomized Kaczmarz (RK) method can thus be described by

$$x_{k+1} = x_k + \frac{b[p(i)] - \langle a_{p(i)}, x_k \rangle}{\|a_{p(i)}\|_2^2} a_{p(i)}, \tag{1.1}$$

where $p(i)$ takes values in $\{1, \ldots, m\}$ with probabilities $\frac{\|a_{p(i)}\|_2^2}{\|A\|_F^2}$. Here and throughout, $\|A\|_F$ denotes the Frobenius norm of A and $\|\cdot\|_2$ denotes the standard Euclidean norm or spectral norm for vectors or matrices, respectively. The selection rule used here is not optimal in general. The motivation for setting the rule according to the weight of the row norms is two-fold. First, it allows for a guarantee of expected exponential convergence for the Kaczmarz method [25]. Second, it is a computationally efficient strategy since often these values will be known approximately or exactly, and will only need to be computed once. A selection rule of this type is of course also related to the idea of preconditioning the matrix A by scaling its rows. Although other diagonal preconditioners may certainly perform better in general, finding such an optimal preconditioner is itself an optimization problem of high complexity. With the above selection strategy, the following exponential bound was shown in [24, 25] for the convergence in expectation of this randomized method:

$$\mathbb{E}\|x_k - x\|_2^2 \leq \left(1 - \frac{1}{R}\right)^k \|x_0 - x\|_2^2, \tag{1.2}$$

where $R = \|A^{-1}\|^2 \|A\|_F^2$ and x_0 is an arbitrary initial estimate. Since we will always assume that A has full column rank, the norm $\|A^{-1}\| \stackrel{\text{def}}{=} \inf\{M : M\|Ax\|_2 \geq \|x\|_2 \text{ for all } x\}$ is well-defined. This bound is essentially independent of the number of rows of A. Moreover, the bound shows that for well conditioned matrices A, the RK method yields expected exponential convergence to the solution in just $\mathrm{O}(n)$ iterations (see Section 2.1 of [25]). Since each iteration consists of a single projection taking $\mathrm{O}(n)$ time, this shows that the overall method has $\mathrm{O}(n^2)$ runtime, which is clearly superior to other methods such as Gaussian elimination which takes $\mathrm{O}(mn^2)$, especially when the system is very large. The discussion in [25] shows that the randomized Kaczmarz method often even outperforms the celebrated conjugate gradient method. For example, when A is a Gaussian matrix and $m > 3n$, the RK method provably requires fewer computations, and empirical studies show that this improvement is substantial. See Section 4.2 of [25] for details.

The empirical and theoretical benefits of this approach lead one to ask whether it is also accurate in the more realistic case when noise is present. One may thus consider the (now possibly inconsistent) system $Ax \approx b + w$ where w is an arbitrary error vector that has been added to the consistent system $Ax = b$. It is shown in [20] that in this case we have exponential convergence to the solution within an error factor:

$$\mathbb{E}\|x_k - x\|_2 \leq \left(1 - \frac{1}{R}\right)^{k/2} \|x_0\|_2 + \sqrt{R}\gamma,$$

where R is the same as above and $\gamma = \max_i \frac{|w[i]|}{\|a_i\|_2}$. It is also shown that this bound is sharp and is attained even for simple examples [20].

1.2 Modified approach

To further improve the convergence rate of the RK method, we suggest a different approach to selecting the rows of A. Although our ideas should also apply seamlessly to the case when noise is present and the system becomes inconsistent, in this work we only consider the noiseless case and leave a detailed analysis in the presence of noise for future work. Since the projections in the algorithm (1.1) are orthogonal, it can be seen that the optimal projection in the kth iteration is the one that maximizes $\|x_{k+1} - x_k\|_2$. By definition of the iterations (1.1), one can calculate these quantities by computing inner products between the rows a_i of A and the current iterate x_k. Since computing one inner product requires $\mathrm{O}(n)$ operations, we clearly cannot afford to perform more than a constant number of these in a given iteration. Our approach, therefore, is to project the rows of A onto a lower dimensional space in such a way that the geometry of the vectors is approximately preserved. We then perform calculations of the form (1.1) with respect to these low dimensional vectors, and select the best projection.

By construction, our modified algorithm will converge to the solution of $Ax = b$ *in the worst case* as fast as the standard RK method. In practice, we expect the convergence to be much faster, especially when the lower dimension d onto which we project the rows is not too small. The improvement in each iteration can be quantified in terms of d and the current estimation x_k, as we demonstrate in Section 3. In Section 4 we demonstrate that empirically our technique outperforms the standard RK method in terms of convergence rate. The runtime of this modified algorithm of course depends on the dimension d onto which we project the rows of A. If $d \ll n$, then each iteration will require $\mathrm{O}(dn)$ operations, meaning that the overall runtime for expected exponential convergence becomes *at most* $\mathrm{O}(dn^2)$. There is a tradeoff in the choice of the dimension d. If d is small, then the runtime per iteration remains small. However, if d is too small then our technique reduces to the standard RK method, and we will not gain in the convergence rate. In the next section we discuss the runtime and implementation, and show why the selection of d on the order of $\log n$ is the right choice. This gives a *worst case* runtime of $\mathrm{O}(n^2 \log n)$, although we expect a much faster convergence as we also see in simulations. This worst case runtime however, is still the same as that of the standard RK method up to the log factor.

2 Implementation and Runtime

Since the projections in the algorithm are orthogonal, one easily sees that the optimal projection in the kth iteration would be the one that maximizes $\|x_{k+1} - x_k\|_2$, or equivalently, the term

$$\frac{|b[i] - \langle a_i, x_k \rangle|}{\|a_i\|_2}. \tag{2.1}$$

Unfortunately, calculating this term takes $\mathrm{O}(n)$ time, so that to keep the overall runtime at $\mathrm{O}(n^2)$ one can only afford to make this computation a constant number of times. However, if we could significantly reduce the dimension of the vectors a_i and x_k used in the calcuation, then more of these calculations could be done at each iteration, and the best out of those computed could be chosen, leading to accelerated convergence. Our idea is thus to project the vectors onto a low dimensional

space such that the geometry is preserved. This will allow approximation of the inner products $\langle a_i, x_k \rangle$ and the norms $\|a_i\|_2$ (if they are not known a priori) from the projected data. To do this, we will consider a Johnson-Lindenstrauss type projection. The well-known Johnson-Lindenstrauss Lemma [17] states that with high probability, there is a projection of a finite set of points onto a space logarithmic in the number of points that approximately preserves geometry. This can be summarized as follows.

Lemma 2.1 (Johnson-Lindenstrauss [17]) *Let $\delta > 0$ and let S be a finite set of points in $\mathbb{R}^n$. Then for any d satisfying*

$$d \geq C \frac{\log |S|}{\delta^2}, \tag{2.2}$$

there exists a Lipschitz mapping $\Phi : \mathbb{R}^n \to \mathbb{R}^d$ such that

$$(1-\delta)\|s_i - s_j\|_2^2 \leq \|\Phi(s_i) - \Phi(s_j)\|_2^2 \leq (1+\delta)\|s_i - s_j\|_2^2, \tag{2.3}$$

for all $s_i, s_j \in S$, where C is an absolute constant.

Remark. The value of C in which this lemma holds depends on the distribution from which Φ is created. When Φ is Gaussian, one has $C \leq 8$ [7].

Although this lemma as stated only guarantees *existence* of such a mapping, in their proof the map Φ is chosen as the projection onto a random d-dimensional subspace of $\mathbb{R}^n$. This result has been improved over time and now one can easily construct such a (random) projection which preserves the geometry (see e.g. [1, 6, 16]). Indeed, it is shown in [1] that whenever a distribution satisfies certain moment conditions, the $d \times n$ random matrix Φ whose entries are chosen i.i.d. with respect to that distribution will satisfy (2.3) with high probability provided d satisfies (2.2). The Guassian distribution, for example, satisfies these moment conditions; therefore the matrix with i.i.d. Gaussian entries will preserve geometry with high probability. Recently there has been work on constructing transforms which satisfy (2.3) but that also provide a fast multiply (see e.g. [2, 15]). For example, Ailon and Chazelle construct in [2] a $C\delta^{-2} \log |S| \times n$ transform Φ satisfying (2.3) with high probability whose multiply requires roughly $n \log n + \delta^{-2} \log^3 |S|$ operations. Hinrichs and Vybiral provide a multiply using $n \log n$ operations when Φ has slightly more rows, on the order of $\log^2 |S|$. Even more recently, Ailon and Liberty show in [3] that when the $d \times n$ Φ is the composition of a randomly subsampled Hadamard (or Fourier) matrix and a random sign matrix, then Φ satisfies (2.3) with high probability when $d = C\delta^{-4} \log |S| \log^4 n$. This matrix has an $n \log n$ multiply, and so this result provides an optimal fast JL transform, up to the power -4 on δ and the polylogarithmic dependence on n.

2.1 Implementation

In our setting, the Johnson-Lindenstrauss Lemma allows us to project the rows of A as well as the estimations x_k onto a space of substantially lower dimension. This will then let us approximately calculate the terms in (2.1) using far fewer operations, which we can use to decide on which hyperplane to project the current estimate. We note that the projection of the rows of A will be performed offline, adding to the preprocessing time, whereas the projection of the estimation x_k will be done at each iteration. We choose to use an analagous strategy as in the RK algorithm for our row selection; that is, using the weight of the row norms. This choice yields a worst case convergence rate which is the same as that guaranteed by the original RK method (see Remark

2 below). This leads to the following modified randomized Kaczmarz method, called Randomized Kaczmarz via Johnson-Lindenstrauss (RKJL) which can be summarized as follows.

Randomized Kaczmarz via Johnson-Lindenstrauss (RKJL)

Input: $m \times n$ matrix A, coefficient vector $b \in \mathbb{R}^m$, parameter d, initial estimate x_0
Output: Approximate x solving $Ax = b$

Initialize: Set $k = 0$, create a $d \times n$ Gaussian matrix Φ and set $\alpha_i = \Phi a_i$. Repeat the following $\mathrm{O}(n)$ times:

Select: Select n rows so that each row a_i is chosen with probability $\|a_i\|_2^2/\|A\|_F^2$ as in 1.1. For each row selected, calculate

$$\gamma_i = \frac{|b[i] - \langle \alpha_i, \Phi x_k \rangle|}{\|\alpha_i\|_2},$$

and set $j = \operatorname{argmax}_i \gamma_i$.

Test: For a_j and the first row a_l selected out of the n, explicitly calculate

$$\gamma_j^* = \frac{|b[j] - \langle a_j, x_k \rangle|}{\|a_j\|_2} \quad \text{and} \quad \gamma_l^* = \frac{|b[l] - \langle a_l, x_k \rangle|}{\|a_l\|_2}.$$

If $\gamma_l^* > \gamma_j^*$, set $j = l$.

Project: Set

$$x_{k+1} = x_k + \frac{b[j] - \langle a_j, x_k \rangle}{\|a_j\|_2^2} a_j.$$

Update: Set $k = k + 1$.

Remarks. 1. We show in Section 3 that $d = \mathrm{O}(\log n)$ will be enough to approximately preserve geometry and thus give convergence improvements. Of course greater values of d may give greater improvements on convergence, at the expense of more computational cost at each iteration.

2. In the Test stage of the algorithm, we see that in addition to approximating the n inner products, we also exactly calculate the inner product of a randomly selected row and also the row that was chosen. This will guarantee that the convergence is not slowed by any drastic consequences of the error in the approximations, and does not affect the overall runtime.

3. We note that the initialization step may of course be computationally expensive, as is calculating the probabilities $p(i)$ in the standard version (1.1). However, this need only be done once, and thus this version of the algorithm will be beneficial for situations in which the same matrix A is used in many problems. This is the case for many applications such as the wave-scattering problem [22] and structural mechanics problems [11]; see [5, 23] for others.

We next turn to an analysis of this modified method. In Section 4 we provide numerical results demonstrating the improved convergence rate.

2.2 Runtime

As discussed above and in [24, 25], the standard randomized Kaczmarz method converges exponentially fast to the solution in $\mathrm{O}(n)$ iterations, resulting in a total runtime of $\mathrm{O}(n^2)$. The RKJL algorithm will thus also converge (in expectation) in at most $\mathrm{O}(n)$ iterations, and so it remains to calculate the runtime of each iteration.

The first step in an iteration is to calculate Φx_k. Since Φ is a $d \times n$ matrix, this computation in general takes $\mathrm{O}(nd)$ time. Next, each α_i lives in a d dimensional space, so that calculating inner products in the selection step costs only $\mathrm{O}(d)$. Since we calculate n such inner products, the total calculation time is $\mathrm{O}(nd)$. The projection and update steps clearly take $\mathrm{O}(n)$ and $\mathrm{O}(1)$ time, respectively, leading to an overall runtime per iteration of $\mathrm{O}(nd)$. Therefore, after $\mathrm{O}(n)$ iterations, we see that the overall runtime of the algorithm is $\mathrm{O}(n^2 d)$. Lemma 3.1 below shows that d can be chosen on the order of $\log n$. Therefore, RKJL converges exponentially fast in at most $\mathrm{O}(n^2 \log n)$ time, which is the same as the runtime of standard RK, up to the log factor. In practice, the runtime is much faster as we show in Section 4.

Because the algorithm will need to use roughly n^2 rows of A (assuming $n^2 \leq m$), the matrix Φ will have to be applied to at least this many vectors, resulting in a $\mathrm{O}(n^3 d)$ initial cost. If the algorithm is used repeatedly for various problem instances over the same matrix A, then one may wish to apply Φ to *all* the rows of A, yielding a $\mathrm{O}(mnd)$ cost. Even when $d = \mathrm{O}(\log n)$ this is of course substantial, but for applications in which the algorithm will be used many times, this one-time cost will become minimal.

This initial computational cost occurs in other methods as well, for example, in submatrix selection algorithms that take $\mathrm{O}(mn^2)$ to select a well represented $n \log n \times n$ submatrix of A (see e.g. [9, 10]). These methods randomly select the submatrix (according to a particular probability model), and if the submatrix selected represents A well, then it can be used to solve the system $Ax = b$. However, there is some probability that the subsystem cannot be solved, in which case it must be reselected again, and so on. This is in contrast to RKJL, for which we are always guaranteed convergence, and most likely with improved expected convergence rate. The choice of method will of course depend on the application, and in some cases it may even be beneficial to use some combination of these approaches.

3 Analytical Justification

We next analyze how the Johnson-Lindenstrauss Lemma is utilized by our method. We will assume here that the system is real-valued and homogeneous (ie. $Ax = 0$), that the rows of A all have unit norm, and that the initial guess x_0 also satisfies $\|x_0\|_2 \leq 1$. These assumptions are of course not necessary, but will make the analysis simpler. We discuss the case where the row norms may be far from equal in Remark 3 below. We begin with an easy lemma which shows that the geometry of the vectors used in the RKJL method is approximately preserved.

Lemma 3.1 *Let Φ be the $n \times d$ (Gaussian) matrix with $d = C\delta^{-2}\log(n)$ as in the RKJL method. Set $\gamma_i = \langle \Phi a_i, \Phi x_k \rangle$ also as in the method. Then $|\gamma_i - \langle a_i, x_k \rangle| \leq 2\delta$ for all i and k in the first $\mathrm{O}(n)$ iterations of RKJL.*

Remark 1 *This lemma shows that with d chosen on the order of $\mathrm{O}(\log n)$, the geometry of the vectors involved in the RKJL method is approximately preserved. In practice, d should thus be chosen of this order to gain improvements in convergence.*

Proof. We employ the Johnson-Lindenstrauss Lemma (Lemma 2.1) with $\mathcal{S}$ consisting of all a_i and x_k in the first $\mathrm{O}(n)$ iterations of RKJL. Then since $|\mathcal{S}| \lesssim n^2$, the condition (2.2) is satisfied, and so (2.3) holds for all a_i and x_k used in the algorithm. By this and the parallelogram law, we have

$$\begin{aligned}
\gamma_i &= \langle \Phi a_i, \Phi x_k \rangle \\
&= \frac{1}{4}\Big(\|\Phi a_i + \Phi x_k\|_2^2 - \|\Phi a_i - \Phi x_k\|_2^2 \Big) \\
&\leq \frac{1}{4}\Big((1+\delta)\|a_i + x_k\|_2^2 - (1-\delta)\|a_i - x_k\|_2^2 \Big) \\
&= \langle a_i, x_k \rangle + \frac{1}{4}\delta(\|a_i + x_k\|_2^2 + \|a_i - x_k\|_2^2) \\
&\leq \langle a_i, x_k \rangle + 2\delta.
\end{aligned}$$

Similarly we have that $\gamma_i \geq \langle a_i, x_k \rangle - 2\delta$, which completes the claim. ■

This shows that the terms γ_i used for selection in the algorithm are approximately equal to the actual desired values $\langle a_i, x_k \rangle$. Thus for δ small, the RKJL algorithm makes well educated decisions at each iteration which allows for quicker convergence (see Theorem 3.2 below). This also shows that when the estimation x_k becomes very close to the true solution $x = 0$, the error δ begins to dominate and improvements may no longer be expected. However, this does not pose a problem since it only occurs when the estimate is already approximately x.

It is clear from construction of the RKJL algorithm (and especially in light of Remark 2 above), that convergence using RKJL is at least as fast as the standard randomized version. Moreover, when the error produced by applying Φ is small, the RKJL method will project onto the "best" hyperplane out of those it selected in that iteration. Since the probability of choosing this "best" row when selecting only a single row is strictly less than the probability of choosing that row when a set of rows is selected, this implies that the only case in which RKJL would not provide a strictly faster convergence rate is when $\langle a_i, x_k \rangle = \langle a_j, x_k \rangle$ for all rows a_i, a_j selected in the kth iteration.

Given a current estimate x_k, one can explicitly compare the expectation of the improvement the next estimation provides, for both the RKJL and standard randomized methods. First observe that if P denotes the projection in the kth iteration, then $x_{k-1} - x_k$ resides in the kernel of P, and is thus orthogonal to the space onto which P projects. This space contains $x_k - x$ since x is the solution to all equations $Ax = b$ and $x_k = Px_{k-1}$. Therefore, $x_k - x$ and $x_{k-1} - x_k$ are orthogonal, implying that

$$\|x_k - x_{k+1}\|_2^2 = \|x - x_k\|_2^2 - \|x - x_{k+1}\|_2^2. \tag{3.1}$$

The relation (3.1) shows that the larger $\|x_k - x_{k-1}\|_2$, the bigger the improvement made in that iteration. We thus fix an estimation x_k and analyze the expectation of $\|x_k - x_{k+1}\|_2$ for the RKJL method versus the standard one. For convenience, we again consider the real and homegenous case (ie. when $b = 0$), and assume the rows of A have unit norms. We then have the following result.

Theorem 3.2 *Fix an estimation x_k and denote by x_{k+1} and x_{k+1}^* the next estimations using the RKJL and the standard RK method, respectively. Set $\gamma_j^* = |\langle a_j, x_k \rangle|^2$ and reorder these so that $\gamma_1^* \geq \gamma_2^* \geq \ldots \geq \gamma_m^*$. Then when $d = C\delta^{-2}\log n$,*

$$\mathbb{E}\|x_{k+1} - x\|_2^2 \leq \min\left[\mathbb{E}\|x_{k+1}^* - x\|_2^2 - \sum_{j=1}^{m} \Big(p_j - \frac{1}{m}\Big)\gamma_j^* + 2\delta, \quad \mathbb{E}\|x_{k+1}^* - x\|_2^2 \right],$$

where

$$p_j = \begin{cases} \frac{\binom{m-j}{n-1}}{\binom{m}{n}}, & j \leq m-n+1 \\ 0, & j > m-n+1 \end{cases}$$

are non-negative values satisfying $\sum_{j=1}^{m} p_j = 1$ *and* $p_1 \geq p_2 \geq \ldots \geq p_m = 0$.

Proof. Since we assume the rows of A have unit norm and that $b = 0$, we see that γ_J is precisely the value of $\|x_{k+1} - x_k\|_2^2$ if the algorithm were to select row J. We begin by examining the kth RKJL iteration.

Let $\mathcal{L}$ denote the set of rows chosen in the selection step, so that $|\mathcal{L}| = n$. If exact geometry were preserved (i.e. $\delta = 0$), then the method would simply select the index of the largest λ_i^* contained in $\mathcal{L}$. However, due to the error induced by Φ, even when the "best" row is selected to be in $\mathcal{L}$, the algorithm may not choose this row for the projection.

We thus define sets T_j for $j = 1, \ldots m$, which consist of rows which could be "confused" in this way,

$$T_j \stackrel{\text{def}}{=} \{i : |\gamma_i^* - \gamma_j^*| \leq 2\delta\}, \quad \text{and} \quad \mu_j = \min\{\gamma_i^* : i \in T_j\}.$$

From Lemma 3.1, if $j \in \mathcal{L}$ and $1, \ldots, j-1 \notin \mathcal{L}$, then the worst row we could choose is one that would give $\|x_{k+1} - x_k\|_2^2 = \mu_j$. Therefore,

$$\begin{aligned} \mathbb{E}\|x_{k+1} - x_k\|_2^2 &= \mathbb{E}_J \gamma_J^* \\ &\geq \mu_1 \mathbb{P}(1 \in \mathcal{L}) + \mu_2 \mathbb{P}(2 \in \mathcal{L} \cap 1 \notin \mathcal{L}) + \ldots + \mu_{m-n+1} \mathbb{P}(1, 2, \ldots, m-n \notin \mathcal{L}) \\ &= \sum_{j=1}^{m-n+1} \mu_j \mathbb{P}(j \in \mathcal{L} \cap 1, \ldots, j-1 \notin \mathcal{L}). \end{aligned}$$

Since each row of A has equal norm, each *row* of A is equally likely to be selected in $\mathcal{L}$. Thus

$$\mathbb{P}(j \in \mathcal{L} \cap 1, \ldots, j-1 \notin \mathcal{L}) = \frac{\binom{m-j}{n-1}}{\binom{m}{n}}.$$

Therefore, we have

$$\mathbb{E}\|x_{k+1} - x_k\|_2^2 \geq \sum_{j=1}^{m-n+1} \mu_j p_j.$$

Lemma 3.1 and the definition of T_j guarantee that $\mu_j \geq \max(\gamma_j^* - 2\delta, 0)$. This along with the fact that $\sum p_j = 1$ yields

$$\mathbb{E}\|x_{k+1} - x_k\|_2^2 \geq \Big(\sum_{j=1}^{m-n+1} \gamma_j^* p_j \Big) - 2\delta. \tag{3.2}$$

Finally, since all the rows of A have the same norm, the standard RK method selects each row uniformly at random, so that

$$\mathbb{E}\|x_{k+1}^* - x_k\|_2^2 = \sum_{j=1}^{m} \frac{1}{m} \gamma_j^*. \tag{3.3}$$

Combining (3.3) with (3.2) and (3.1) we have

$$\begin{aligned}\mathbb{E}\|x_{k+1}-x\|_2^2 &= \mathbb{E}\|x_k-x\|_2^2 - \|x_k-x_{k+1}\|_2^2\\ &\le \mathbb{E}\|x_k-x\|_2^2 - \Big(\sum_{j=1}^{m-n+1}\gamma_j^* p_j\Big) + 2\delta\\ &= \mathbb{E}\|x_{k+1}^*-x\|_2^2 + \mathbb{E}\|x_{k+1}^*-x_k\|_2^2 - \Big(\sum_{j=1}^{m-n+1}\gamma_j^* p_j\Big) + 2\delta\\ &= \mathbb{E}\|x_{k+1}^*-x\|_2^2 - \sum_{j=1}^{m}\Big(p_j-\frac{1}{m}\Big)\gamma_j^* + 2\delta.\end{aligned}$$

This along with the fact that by construction of RKJL, $\mathbb{E}\|x_{k+1}-x\|_2^2 \le \mathbb{E}\|x_{k+1}^*-x\|_2^2$, completes the claim. ■

Remarks. **1.** Theorem 3.2 gives a lower bound, which shows improvements in the "worst case", when the error induced by the Johnson-Lindenstrauss projection causes the method to choose a row of A in the worst way. Numerical experiments (as seen in the next section) demonstrate substantial improvements in the convergence rate.

2. Note that since the sequences $\{\gamma_j^*\}$ and $\{p_j\}$ are non-increasing and $\sum_{j=1}^m p_j = 1 = \sum_{j=1}^m \frac{1}{m}$, the sum $\beta = \sum_{j=1}^m \Big(p_j - \frac{1}{m}\Big)\gamma_j^*$ is non-negative. Furthermore, $\beta = 0$ only when $\gamma_1^* = \gamma_2^* = \ldots = \gamma_m^*$. Indeed, if $\gamma_i^* > \gamma_j^*$ even for just one pair $i < j$, then for δ small enough, the RKJL method provides strict convergence improvement. Knowledge of x_k and A would allow one to precisely calculate the improvement.

3. The theorem is proven under the assumption that the rows of A have the same norm. The same argument holds (with different values of p_j) still showing improvement when this assumption does not hold, and numerical experiments show similar results in either case. Although the analysis of exact improvement in these other cases may quickly become quite complicated, we recall that by construction the RKJL method offers overall improvement in any case.

It is also helpful to identify some particularly interesting scenarios, for example, when one or a few rows are substantially of larger norm. In this case the standard RK algorithm (noticing that each row is selected independently of the previous selections) will choose this row repeatedly with high probability. Clearly this may slow convergence (especially when these rows are highly correlated), and although there is still guaranteed exponential convergence, R in this case is much larger so that the guaranteed rate is slower. In RKJL, although the guaranteed worst case rate is the same, these large rows are likely not to be selected when their contributions toward the solution is minimal. This will again speed up convergence, since in the language of Theorem 3.2, the λ_k^* corresponding to the rows which are highly correlated with the previous projection will be such that $k \approx m$. This means that the same argument as in Theorem 3.2 will hold for all λ_j^* with $j \le k \approx m$. Although this means the improvement in RKJL may not be as substantial as in the case of equal normed rows, we will still see a large improvement.

These ideas highlight the fact that the selection strategy is not optimal in general. For example, if there are $k \ll m$ rows which are highly uncorrelated but have very small norm, and $m - k \approx m$ equal rows with substantially larger norm, neither RK nor RKJL will perform well. Of course if this were reversed, with the uncorrelated rows having large norm and the equal rows having small norm, the selection strategy will yield excellent convergence in both RK and especially

RKJL. We emphasize again that the choice of the selection rule is certainly not optimal in general, but is computationally efficient and allows for provable expected exponential rate of convergence. Choosing an optimal selection strategy for a given system is itself a problem of high complexity. If one is using RKJL and investing preprocessing time to perform dimension reduction, one may also wish to simply normalize the rows to avoid such difficulties.

Theorem 3.2 implies the following corollary, showing the improved convergence using RKJL when exact geometry is preserved (i.e. when $\delta \to 0$).

Corollary 3.3 *Fix an estimation x_k and denote by x_{k+1} and x^*_{k+1} the next estimations using the RKJL and the standard method, respectively. Set $\gamma^*_j = |\langle a_j, x_k\rangle|^2$ and reorder these so that $\gamma^*_1 \geq \gamma^*_2 \geq \ldots \geq \gamma^*_m$. Then when exact geometry is preserved ($\delta \to 0$),*

$$\mathbb{E}\|x_{k+1} - x\|_2^2 \leq \mathbb{E}\|x^*_{k+1} - x\|_2^2 - \sum_{j=1}^{m} \Big(p_j - \frac{1}{m}\Big)\gamma^*_j.$$

4 Numerical Results

We now demonstrate improved convergence using the RKJL method. The first experiment we run is in the computationally infeasible situation where we do not use the Johnson-Lindenstrauss projection, but simply choose the best row out of the randomly selected n rows. This experiment will demonstrate the improved convergence using RKJL when the error δ induced by Φ goes to 0. This is the best improvement one can hope for in RKJL. When the problem sizes grow very large, the effect of the δ^2 term in (2.2) becomes minimal, so it may be realistic to take δ quite small. For these simulations we use a 60000×1000 matrix with Bernoulli entries and use a homogeneous system with an initial estimate chosen uniformly at random on the sphere. We see in Figure 1 that the convergence in this scenario is significantly improved.

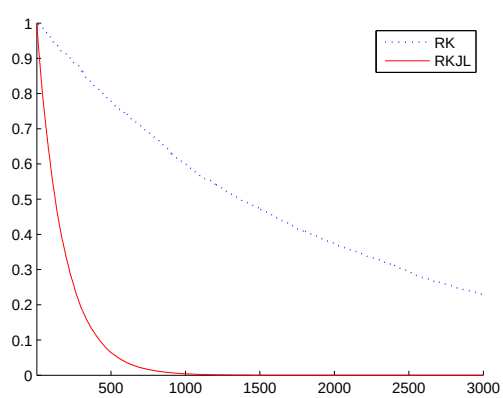

Figure 1: ℓ_2-Error (y-axis) as a function of the iterations (x-axis). The dashed line is standard Randomized Kaczmarz, and the solid line is the modified one, without a Johnson-Lindenstrauss projection. Instead, the best move out of the randomly chosen n rows is used. Note that we cannot afford to do this computationally.

Next we run simulations using a Johnson-Lindenstrauss matrix Φ. We generate Φ and the system $Ax = b$ the same as above, but now run RKJL using various values of d. In Figure 2 we see exactly what we expect, that with higher values of d (corresponding to lower δ values), we have much quicker convergence. The speedup in convergence using larger d needs to be weighed against the increase in computation per iteration, as was discussed in Section 2. Since right now there are no theoretical guarantees on precisely how the convergence is affected by larger d, this comparison should be done empirically. Finally, it is clear that as m and n grow large, the impact on d of forcing δ to be small becomes minimal. Thus for very large systems, using $d = \mathrm{O}(\log n)$ will give convergence that looks more like that in Figure 1.

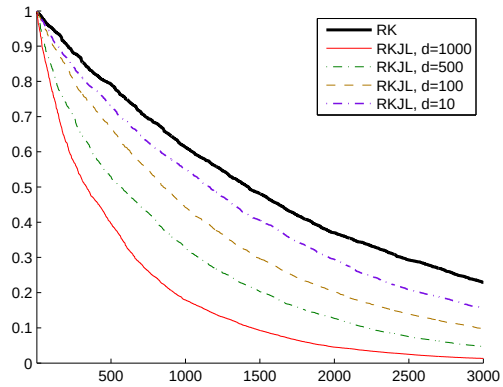

Figure 2: ℓ_2-Error (y-axis) as a function of the iterations (x-axis) for various values of d with $m = 60000$ and $n = 1000$.

Acknowledgements

This work is partially supported by the NSF DMS EMSW21-VIGRE grant and the Israel Science Foundation under Grant no. 1081/07. We would also like to thank Emmanuel Candès and Thomas Strohmer for helpful suggestions.

References

[1] D. Achlioptas. Database-friendly random projections: Johnson–Lindenstrauss with binary coins. *J. Comp. Sys. Sci.*, 66(4):671–687, 2003. Special issue of invited papers from PODS'01.

[2] N. Ailon and B. Chazelle. The fast johnson-lindenstrauss transform and approximate nearest neighbors. *SIAM J. Comput.*, 39:302–322, 2009.

[3] N. Ailon and E. Liberty. Almost optimal unrestricted fast johnson-lindenstrauss transform. Available at http://arxiv.org/abs/1005.5513, 2010.

[4] C. Cenker, H. G. Feichtinger, M. Mayer, H. Steier, and T. Strohmer. New variants of the POCS method using affine subspaces of finite codimension, with applications to irregular sampling. *Proc. SPIE: Visual Communications and Image Processing*, pages 299–310, 1992.

[5] T. F. Chan and W. L. Wan. Analysis of projection methods for solving linear systems with multiple right-hand sides. *SIAM J. Sci. Comput.*, 18:1698–1721, 1997.

[6] D. Dasgupta and A. Gupta. An elementary proof of the JohnsonLindenstrauss lemma. Tech. Report 99-006, U.C. Berkeley, 1999.

[7] S. Dasgupta and A. Gupta. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65, 2003.

[8] F. Deutsch and H. Hundal. The rate of convergence for the method of alternating projections. *J. Math. Anal. Appl.*, 205(2):381–405, 1997.

[9] P. Drineas, M.W. Mahoney, and S. Muthukrishnan. Sampling algorithms for ℓ_2 regression and applications. In *Proc. 17th Ann. ACM-SIAM Symp. on Discrete Algorithms*, pages 1127–1136, 2006.

[10] P. Drineas, M.W. Mahoney, and S. Muthukrishnan. Relative-error CUR matrix decompositions. *SIAM J. Matrix Anal. A.*, 30:844–881, 2008.

[11] C. Farhat and F. X. Roux. Implicit parallel processing in structural mechanics. Tech. report cu-cssc-93-26, Center for Aerospace Structures, University of Colorado, Boulder, CO, 1993.

[12] A. Galàntai. On the rate of convergence of the alternating projection method in finite dimensional spaces. *J. Math. Anal. Appl.*, 310(1):30–44, 2005.

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

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

[15] A. Hinrichs and J. Vybiral. Johnson-lindenstrauss lemma for circulant matrices. submitted, 2009.

[16] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Symp. on Theory of Computing*, pages 604–613, 1998.

[17] W. B. Johnson and J. Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Conf. in Modern Analysis and Probability*, pages 189–206, 1984.

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

[19] F. Natterer. *The Mathematics of Computerized Tomography.* Wiley, New York, 1986.

[20] D. Needell. Randomized Kaczmarz solver for noisy linear systems. *BIT Num. Math.*, 2010.

[21] K. M. Sezan and H. Stark. Applications of convex projection theory to image recovery in tomography and related areas. In H. Stark, editor, *Image Recovery: Theory and application*, pages 415–462. Acad. Press, 1987.

[22] C. F. Smith, A. F. Peterson, and R. Mittra. A conjugate gradient algorithm for the treatment of multiple incident electromagnetic fields. *IEEE Trans. Antennas and Propagation*, 37:1490–1493, 1989.

[23] T. Strohmer. A levinson-galerkin algorithm for regularized trigonometric approximation. *SIAM J. Sci. Comput.*, 22(4):1160–1183, 2000.

[24] T. Strohmer and R. Vershynin. A randomized solver for linear systems with exponential convergence. In *RANDOM 2006 (10th International Workshop on Randomization and Computation)*, number 4110 in Lecture Notes in Computer Science, pages 499–507. Springer, 2006.

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