Published in *Numerical Algorithms* 77:4, 2018, 1141–1157
https://link.springer.com/article/10.1007/s11075-017-0356-3

A derandomization approach to recovering bandlimited signals across a wide range of random sampling rates

Dan Gordon*

Abstract

Reconstructing bandlimited functions from random sampling is an important problem in signal processing. Strohmer and Vershynin obtained good results for this problem by using a randomized version of the Kaczmarz algorithm (RK) and assigning to every equation a probability weight proportional to the average distance of the sample from its two nearest neighbors. However, their results are valid only for moderate to high sampling rates; in practice, it may not always be possible to obtain many samples. Experiments show that the number of projections required by RK and other Kaczmarz variants rises seemingly exponentially when the equations/variables ratio (EVR) falls below 5. CGMN, which is a CG acceleration of Kaczmarz, provides very good results for low values of EVR and it is much better than CGNR and CGNE. A derandomization method, based on an extension of the bit-reversal permutation, is combined with the weights and shown to improve the performance of CGMN and the regular (cyclic) Kaczmarz, which even outperforms RK. A byproduct of our results is the finding that signals composed mainly of high-frequency components are easier to recover.

Keywords. *Bandlimited functions; bit-reversal; CGMN; derandomization; extended bit-reversal; low sampling rates; randomized Kaczmarz; RK; signal processing.*

1 Introduction

Recovery of bandlimited functions from random sampling is an important problem in signal processing; see, for example [23, 28, 29]. One of the iterative approaches that has been used is the Kaczmarz algorithm (KACZ) [21], which is a very well known solution method for linear systems in various applications, such as computerized tomography (CT), where it also known as ART (algebraic reconstruction technique) [18]. For additional background on iterative algorithms in signal processing, see [3, §7].

KACZ is best described by its simple geometric explanation: starting from some selected point in the solution space $\mathbb{R}^n$ or $\mathbb{C}^n$, the current iterate is repeatedly projected orthogonally onto the

*Dept. of Computer Science, University of Haifa, Haifa 34988, Israel. Email: gordon@cs.haifa.ac.il

hyperplane defined by one of the system's equations. Usually, the projections follow cyclically the given order of the linear system. It is also known that for some applications, a random selection of the equations can provide better results than the cyclic order [4, 19].

We consider an $m \times n$ linear system

$$Ax = b, \tag{1}$$

where $b = (b_1, \ldots, b_m)^T$, and the jth row of A is denoted by a_j. Denote by $x^0, x^1, \ldots$ the sequence of KACZ iterates, and suppose that x^{k+1} is obtained from x^k by projecting x^k onto the jth hyperplane, i.e.,

$$x^{k+1} = x^k + \lambda \frac{b_j - \langle a_j, x^k \rangle}{\|a_j\|_2^2} a_j, \tag{2}$$

where $0 < \lambda < 2$ is a relaxation parameter. We can save some computation time by initially dividing each equation $\langle a_j, x \rangle = b_j$ by $\|a_j\|_2$, thus avoiding the division by $\|a_j\|_2^2$ at every step; this is sometimes called normalizing the equations.

Strohmer and Vershynin [28] consider a randomized version of KACZ, called RK, where equation j is selected with probability proportional to $\|a_j\|_2^2$. They prove that RK converges at an exponential rate by estimating the expected error after k iterations in terms of the scaled condition number of A [9] $\kappa(A)$. This result is applied to the problem of reconstructing bandlimited signals from random sampling. Every sample provides one equation, and, based on a concept of Feichtinger and Gröchenig [11], the equation is assigned a probability weight that is proportional to the average distance of the sample from its two nearest neighbors. For convenience, we refer to these weights as the FG-weights. In one example in [28], RK is applied to a system with $m = 700$ equations and $n = 101$ variables: every equation is multiplied by the square root of its corresponding weight, and in RK, the equation is selected with probability proportional to its weight. Very good results were obtained this way.

However, it is clear from the geometric nature of KACZ and from Eq. (2) that any nonzero scaling of the equations has no effect, as the scaling factor cancels out. This was pointed out by Censor, Herman and Jiang [6]. It follows that the good results obtained with RK are due to the random selection according to the FG-weights.

Several authors extended [28] in various directions. Needell [25] extended the results to modeling noise in the sample readings, and Eldar and Needell [10] used dimension reduction techniques to speed up the calculations on strongly overdetermined systems. More recently, Liu and Wright [22] presented an accelerated RK algorithm (ARK), based on a method due to Nesterov [26].

In this paper we enlarge on this topic and study the application of several Kaczmarz-based methods to the problem of reconstructing bandlimited signals from random sampling for a wide range of the equations/variables ratio (EVR). We consider both cyclic and random versions

of Kaczmarz, as well as three CG methods, including CGMN [1]. For cyclic Kaczmarz and CGMN, we introduce a "shuffled" order of the equations that is based on an extension of the well-known bit-reversal permutation; this is referred to as EBR (extended bit-reversal). We also augment EBR with some of the equations with the largest FG-weights (EBRW) and obtain even better results.

Our main results can be summarized as follows:

1. For EVR < 5, the number of projections required by all Kaczmarz variants to achieve a given relative residual rises very steeply, thus rendering these methods virtually useless below this threshold.
2. For low ($\sim$2) to moderate (6-6.5) values of EVR, CGMN, CGNR and CGNE are the only viable alternatives. CGMN is much more efficient that CGNR and CGNE, particularly when applied to the EBRW-ordered equations.
3. For moderate to high values of EVR, the original (cyclic) Kaczmarz on the EBRW-ordered equations achieves the best results.
4. The worst mode for (cyclic) Kaczmarz and CGMN is to sort the equations according to the values of their corresponding samples. This problem will be explained in §2.1.
5. It is easier to recover a signal that is composed mainly of high-frequency components than a signal with the same number of components but at lower frequencies.
6. The "accelerated RK" (ARK) [22] exhibits a typical Kaczmarz-based behavior, but it is not competitive with the other Kaczmarz variants on this problem.
7. A randomized subset version of the Cimmino algorithm [7] also behaves similarly to the Kaczmarz variants, but it is also not competitive.

The following sections enlarge on the various solution methods that are used and set up the problem to be solved. This is followed by the experimental results and the conclusions.

2 Algorithms and derandomization

2.1 Kaczmarz and variants

As mentioned previously, the original Kaczmarz algorithm starts from some selected point and successively projects the current iterate onto the hyperplane determined by the next equation, in cyclic order. To avoid confusion between the various Kaczmarz variants, we shall refer to this as C-KACZ (cyclic Kaczmarz). In [28, §4.1] it is mentioned that the sample points are sorted in ascending order after they are selected. This means that we can now have two orders of the equations: the random order corresponding to the (original) random order of the samples, and the order corresponding to the sorted order of the samples.

C-KACZ on the sorted equations behaves very differently from all other Kaczmarz variants, due to the following well known problem, which was also noted by Mayer in [24]. Consider the simple case shown in Fig. 1: two equations in two variables, with a very small angle between their normals. Regardless of the initial starting point, the sequence of iterates quickly settles into a pattern that converges to the solution very slowly. In the case of the equations sorted according to their corresponding samples, the values of two successive samples are close, resulting in successive equations with a small angle between their normals. So, the more samples we have, the smaller the angles between successive hyperplanes. We shall see the effect of this on the behavior of C-KACZ on the sorted equations.

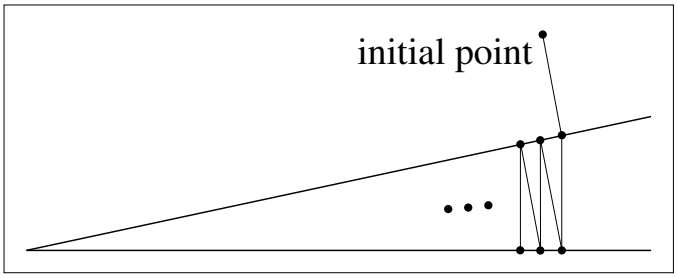

Figure 1: Successive Kaczmarz projections between two lines with a small angle between them.

We will use the generic term R-KACZ for randomized Kaczmarz, of which we have two versions: R-KACZ-SR, in which the equations are selected with simple randomization, and R-KACZ-FG, where equations are selected according to the FG-weights.

2.2 CG methods

The Conjugate Gradients algorithm (CG) [20] and its variants, as well as other Krylov subspace methods, are very well known and used extensively in numerical computations. For non-symmetric matrices, the CGNR and CGNE methods can be used; see [27]. CGNR is the application of CG to the system $A^*Ax = A^*b$, where A^* is the conjugate transpose of A. CGNR is also known as CGLS. Also, CGNE is the application of CG to the system $AA^*y = b$, $x = A^*y$; note that Kaczmarz is the application of SOR (successive over-relaxation) to the latter system [1].

Since the equations are normalized for the efficiency of the Kaczmarz methods, CGNR and CGNE were also applied to the normalized equations. It turns out that CGNR on the normalized equations is actually a CG acceleration of the Cimmino algorithm [7], which is also a projection technique; this was pointed out in [15, p. 30]. It has also been shown in [15] and related works that CGNR on the normalized equations is much more efficient than CGNR by itself when the equations are ill-conditioned due to having many large off-diagonal elements, and/or in the case of discontinuous coefficients.

In a landmark paper, Björck and Elfving [1] developed a CG acceleration of Kaczmarz, called CGMN. Briefly, CGMN can be explained as follows: by applying C-KACZ once to the system of equations, followed by applying it to the equations in reverse order, one obtains a symmetric and positive semi-definite iteration matrix, which we denote by B. CGMN is the application of CG to a certain derived linear system with the matrix B. However, B is not computed; instead, matrix-vector computations with B are done by running Kaczmarz forward and backward on the system of equations. CGMN has been used very successfully on linear systems derived from convection dominated partial differential equations (PDEs); see [14] and its references.

For huge problems which require parallel processing, there is also a block-parallel version of CGMN, called CARP-CG [16]. CARP-CG is a CG acceleration of CARP [13], which is a block-parallel version of the Kaczmarz algorithm. CARP and CARP-CG were developed as a type of domain decomposition technique for the solution of linear systems derived from PDEs. CGMN and CARP-CG combine the inherent robustness of Kaczmarz with the efficiency of CG. More relevant to signal processing is the fact that CARP-CG was found to be particularly adept at solving the difficult numerical problem of the Helmholtz equation at high frequencies – see [17] and its references. CARP is also a "string-averaging" technique according to the general definition of this term in [5]. However, there is no actual averaging of intermediate results in CARP; instead, partial results in different subdomains are combined by averaging only variables that belong to two or more subdomains.

2.3 Derandomization: the extended bit-reversal permutation

If we want to accelerate KACZ using CGMN, we must use some cyclic order of the equations. The sorted (according to the samples) order is not suitable due to the small angles between successive hyperplanes. We can use the original random order of the samples, but this does not necessarily eliminate all the problematic cases, especially when the number of samples is large. We therefore introduce a certain "shuffling" order and apply it to the sorted equations. An ideal shuffle would be the bit-reversal permutation, which is well-known in the signal processing field as it is implicit in the recursive FFT (fast Fourier transform) [8]. Consider the sequence of numbers $0,1,\ldots,7$, and suppose we change the position of every number to the position obtained when its base-2 representation is reversed. E.g., the number 3, whose binary representation is 011, is moved to the position 6, whose binary representation is 110. We then get the sequence $0,4,2,6,1,5,3,7$. This way, adjacent numbers are separated. For another reordering approach to C-KACZ, see [18, p. 209].

The bit-reversal permutation is defined only when the size m of the sequence is a power of 2, so in order to use it on arbitrarily-sized sequences, we extended it to allow sequences of any size. We call this the "extended bit-reversal" (EBR) permutation. This should not be confused

with the term "generalized bit-reversal", which refers to the use of other radix bases, and also to mixed radix representations. EBR operates as follows: it is first applied recursively to the lower half of the sequence, and the positions the elements are doubled. Then, vacant positions are filled with the rest of the elements. For sequences of odd length, the middle element is placed at the center and the other elements skip over the central position. The following pseudo-code details these operations.

begin algorithm EBR

EBR(array $a[\cdot]$, size n) // recursive routine

// Action: $a[0]\cdots a[n-1]$ are filled with the numbers

// $0,\ldots,n-1$, ordered by the EBR permutation

if $(n=1)$ $a[0]:=0$; **return**

$k:=\lfloor n/2\rfloor$

// define a shift operator on indices to skip the central position when n is odd:

define $S(i)=($**if** $(i<k)$ **then** i **else** $i+1)$

EBR($a[\cdot]$, k) // recursive call with half the array size

if (n is even)

for $(i=k-1,k-2,\ldots,1)$ // double the positions of the recursive result

$a[2i]:=a[i]$; $i:=i-1$

for $(i=1,3,\ldots,n-1)$ // fill the empty spaces

$a[i]:=a[i-1]+k$; $i:=i+2$

else // n is odd – use the shift operator $S(i)$

for $(i=k-1,k-2,\ldots,1)$ // double the positions of the recursive result

$a[S(2i)]:=a[i]$; $i:=i-1$

for $(i=1,3,\ldots,n-2)$ // fill the empty spaces

$a[S(i)]:=a[S(i-1)]+k+1$; $i:=i+2$

$a[k]:=k$ // place the middle number in the central position

return

end algorithm EBR

Note that the above formulation works on sequences starting from 0. Since the recursion is only done on half the array, the complexity of EBR is linear in n. When n is a power of 2, the result is the standard bit-reversal permutation. EBR separates adjacent elements quite well: it can be shown that for $k\geq 2$ and $2^k\leq n<2^{k+1}$, adjacent elements in a sequence of length n will be separated in the permuted sequence by at least 2^{k-2} positions.

By applying EBR to the equations sorted according to the samples, we avoid the problematic adjacency of equations with a small angle between their normals. However, using only the EBR-shuffled equations ignores the advantage of utilizing the FG-weights. We therefore added to the m EBR-shuffled equations a fraction $0<W<1$ of the equations with the largest FG-

weights. Based on our experiments, we set $W = \min(0.2, n/m)$ and added $k = \lfloor Wm \rfloor$ equations, so the total number of equations was $m+k$. Finding the k largest weights can be done in linear time by first finding the kth largest weight (w_k) in linear time using the well-known selection algorithm of Blum et al. [2], and then finding $k-1$ weights that are larger than w_k. The k equations corresponding to these weights are then appended to the list of the EBR-shuffled equations. The combined list, called the *EBRW-ordered equations*, will be used with C-KACZ and CGMN.

2.4 Other tested methods

Liu and Wright [22] present an acceleration of RK, called ARK. As mentioned in the Introduction, this method uses an acceleration technique of Nesterov [26] to speed up RK. In [22, §6.3] the authors write that they do not consider the case $m \gg n$ because all methods converge rapidly on such cases. However, we also implemented this method for comparison, together with an FG-weighted version (ARK-FG).

The above-mentioned Cimmino algorithm [7], which we will denote by CIMM, is also based on orthogonal projections. However, instead of sequential projections, all the projections are done from the current iterate and averaged to form the next iterate. Assuming the equations have been normalized, a typical step of CIMM is the following:

$$x^{k+1} = x^k + \frac{\lambda}{m} \sum_{j=1}^{m} \left(b_j - \langle a_j, x^k \rangle a_j \right), \tag{3}$$

where $\lambda > 0$ is a relaxation parameter. It was mentioned earlier that CGNR, applied to the normal equations, is a CG acceleration of CIMM; however, λ cancels out in this derivation.

In the present problem, there are many pairs of hyperplanes with very small angles between their normals, and this requires a very large value for λ. An example can be seen in Fig. 2 below, which shows two lines, L_1, L_2 forming a small angle, and a point x_1 between them. x_1 is projected orthogonally towards the lines with a large value of λ, forming the two points y_1, y_2, whose average x_2 is the next iterate, which is also the intersection of L_1 and L_2. In the general case, there is always an optimal value of λ which provides the best rate of convergence.

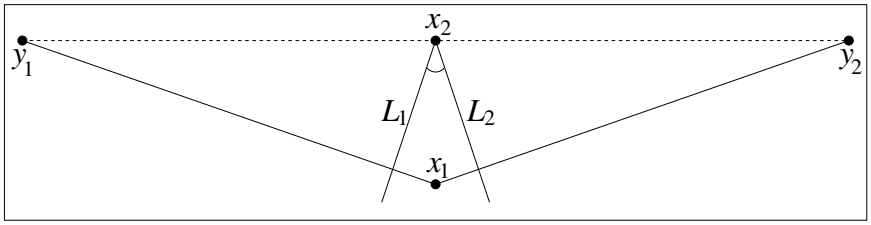

Figure 2: One step of CIMM with a large relaxation parameter produces the solution.

We have implemented a randomized version of CIMM, called R-CIMM, which operates as follows on the normalized system (1):

begin algorithm R-CIMM

select the following 3 items:

an integer $1 \leq s \leq m$, a relaxation parameter λ, a starting point x^0.

repeat for $k = 0, 1, 2, \ldots$ until convergence:

randomly select a subset $S \subseteq \{1, \ldots, m\}$ of size s.

set $x^{k+1} = x^k + \dfrac{\lambda}{s} \displaystyle\sum_{j \in S} \Big(b_j - \langle a_j, x^k \rangle a_j \Big)$.

end algorithm R-CIMM

For the problem considered in this paper, the random selection was made according to the FG-weights, and the resulting algorithm is denoted by R-CIMM-FG. Note that for $s = 1$, R-CIMM-FG is identical to R-KACZ-FG.

3 Experimental results

3.1 Setup of the experiments

We consider the experimental problem set out in [28, §4.1]: A bandlimited function $f(t)$ is known to be of the form

$$f(t) = \sum_{\ell=-r}^{r} x_\ell e^{2\pi i \ell t} \tag{4}$$

with $x_\ell \in \mathbb{C}$. r is assumed to be known, and we are given the values of $f(t_j)$ for $t_j \in \mathbb{R}$, $j = 1, \ldots, m$, and $m \geq n$, where $n = 2r + 1$. Our problem is to determine $x_1, \ldots, x_n$.

In many practical cases, the samples are not given uniformly randomly, so the sampling is usually called "irregular sampling". However, for purposes of comparison, our experiments followed the methodology of [28, p. 270], in which the samples were selected uniformly randomly. Let $t_1, \ldots, t_m$ be the random samples selected uniformly from the interval [0, 1]. We denote the complex elements $f(t_j)$ as $f(t_j) = u_j + iv_j$, and select u_j and v_j uniformly randomly from $[-1, 1]$. Thus, for each $1 \leq j \leq m$, we get one complex equation in the variables $x_1, \ldots, x_n$, with the RHS being the known value of $f(t_j)$. We assume that the system of equations is now as given by Eq. (1).

The FG-weights associated with the samples are defined as follows. Assume that the samples are sorted in increasing order, so that the order $t_1, \ldots, t_m$ is the actual order of the samples on

the real line. The FG-weights w_j associated with t_j for $1 \leq j \leq m$ are then defined by:

$$\begin{aligned} w_1 &= t_1 + (t_2 - t_1)/2, \\ w_m &= 1 - t_m + (t_m - t_{m-1})/2, \\ w_j &= (t_{j+1} - t_{j-1})/2, \ \text{ for } \ 1 < j < m. \end{aligned} \tag{5}$$

w_j can be viewed as a measure of the "isolation" of t_j from its nearest neighbors, so the random selection of the equations according to the FG-weights means that the more isolated a sample is, the more likely the corresponding equation will be used in the R-KACZ-FG process.

3.2 Main results

Similarly to the experiment of [28], we start with the same number of variables – 101. However, whereas [28] considered only the case of 700 equations, we expand the range from 202 equations to 1500 equations, and show the number of projections required by the Kaczmarz-based methods (including CGMN) to reach a relative residual of 10^{-14}. This is shown in Fig. 3. All the points on the plots are obtained from the average of at least 100 experiments. Not shown in the figure is the plot of C-KACZ on the sorted equations; its lowest point is at 500 equations, where it takes an average of 58,230 projections to reach the relative residual goal. The plots start at 202 equations because convergence to the required goal is problematic with fewer equations.

Generally, we can see from Fig. 3 that the Kaczmarz variants show a very steep increase in the number of projections below 500 equations, whereas CGMN (except for the sorted equations case) handles this range quite well. It can be seen that there are three ranges of data:

- the low range, which is handled best by CGMN on the EBRW-ordered equations;
- a small central region around 600 equations in which three methods produce similar results, and two KACZ methods overtake CGMN on the EBRW-ordered equations;
- the high range, which is handled best by C-KACZ on the EBRW-ordered equations.

The reason for these three ranges is the interplay of the various factors affecting the rate of convergence. At the low range, the CG acceleration of Kaczmarz is the most effective, with the EBRW-ordering giving CGMN an edge over the random ordering of the equations. The significance of the FG-weights is most effective at the low and middle ranges.

However, as the number of equations increases, the samples become more dense, and this has two effects. Firstly, samples generally become less and less "isolated", thus reducing the effect of the FG-weights. Secondly, the problematic issue of equations with small angles between the normals of their hyperplanes becomes more severe, thereby giving an advantage to shuffling methods that avoid the proximity of such pairs of equations. So, at some point, C-KACZ on the EBRW-ordered equations takes the lead. Note that by selecting $W = \min(0.2, n/m)$, we add

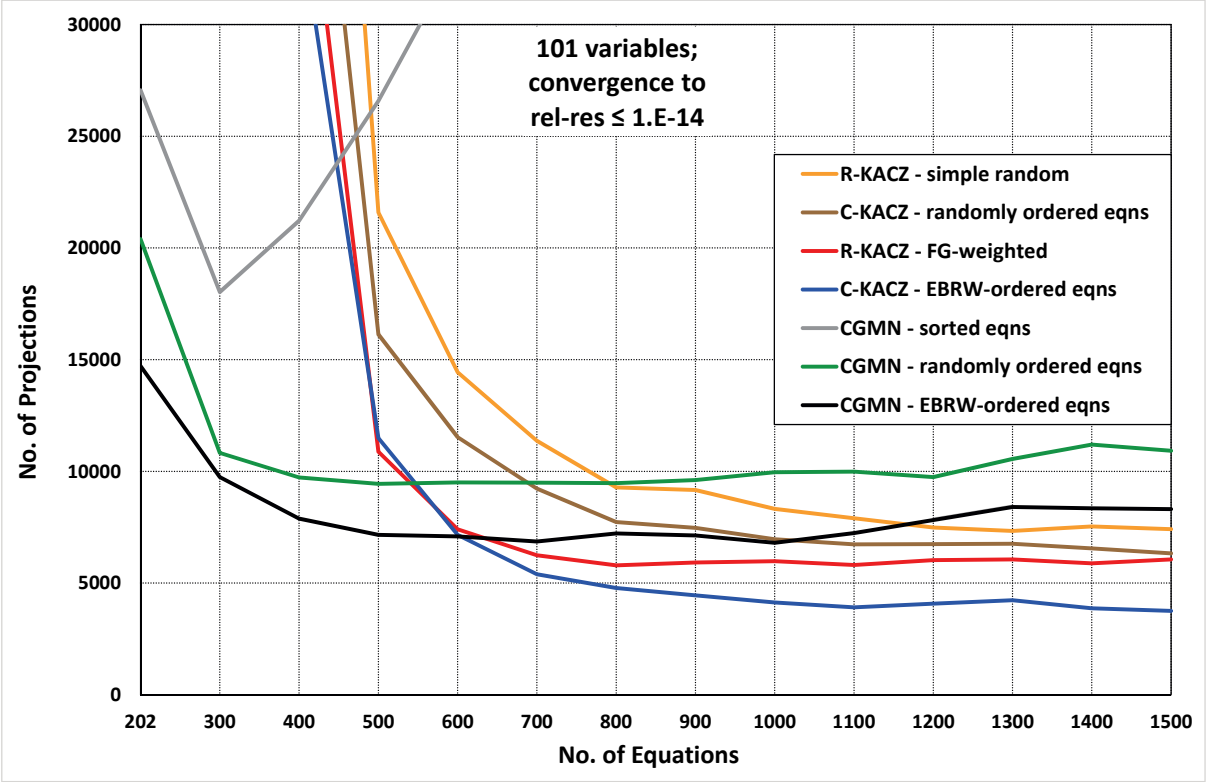

Figure 3: Plot of no. of projections to obtain a relative residual of at least 10^{-14} vs. no. of equations, for 101 variables. C-KACZ is cyclic Kaczmarz, R-KACZ is randomized Kaczmarz.

20% to the number of equations for EVR<5, but starting from EVR=5, the number of additional equations decreases according to the diminishing significance of the FG-weights.

The reason why C-KACZ overtakes CGMN on the EBRW-ordered equations at some point is the following: CGMN requires one double KACZ sweep to initialize the CG process, and then, every CG iteration also requires a double sweep. However, C-KACZ does not need an initial double sweep and it can also stop with an odd number of sweeps.

Consider the odd behavior of CGMN on the sorted equations: the number of projections decreases and then increases. This is due to the fact that at low EVR values, the small angles between equation normals are not yet so effective, but as the number of equations (and samples) increases, the effect of the many adjacent small angles becomes detrimental for CGMN, and even more so for C-KACZ.

The results pose an intriguing problem: what is the reason for the sudden steep rise in the number of projections required by the Kaczmarz variants below 500 equations? This rise actually starts somewhere between 500 and 800 equations, depending on the particular variant of Kaczmarz being used. It is tempting to conjecture that the problem lies with the high-frequency components, but the following experiments indicate the opposite. We repeated the main experiments with the same number of equations (101), but now instead of frequencies in the range

$-50 \leq k \leq 50$, we set $k \in \{0, \pm 50, \pm 51, \ldots, \pm 100\}$. The results, shown in Fig. 4, indicate that now, the Kaczmarz variants can handle systems with less than 500 equations, and they also converge much faster than previously on strongly overdetermined systems. However, the relative performance of the different methods is similar to the previous cases.

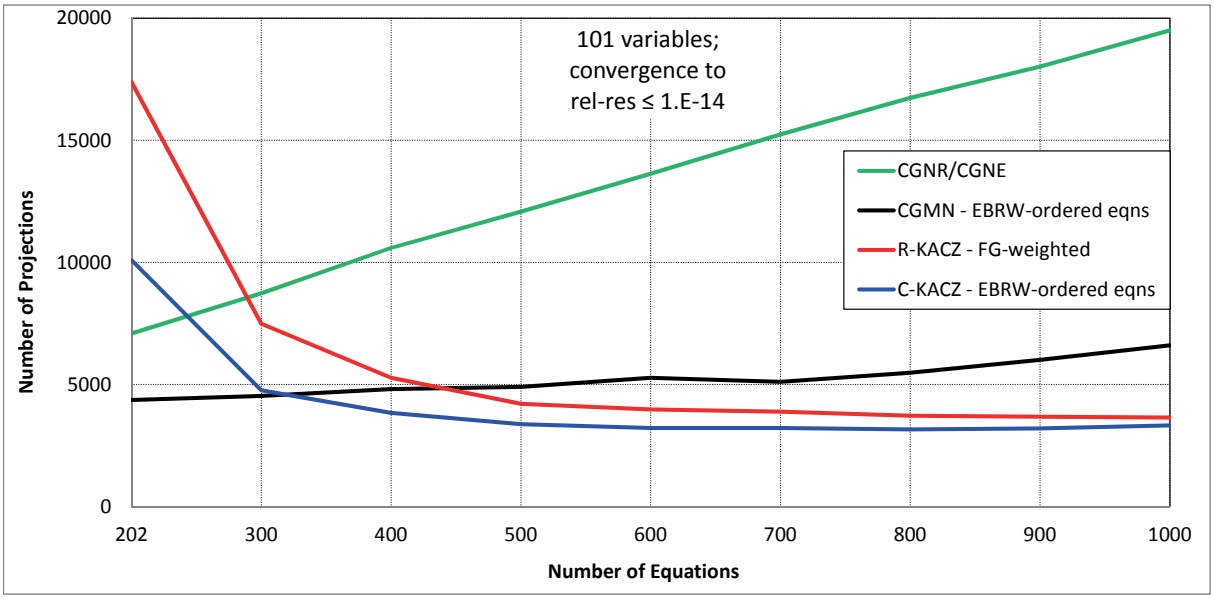

Figure 4: Plot of no. of projections to achieve a relative residual of 1.E-14 on a system with 101 equations with mostly high-frequency components, using randomized KACZ with FG-weights, cyclic KACZ and CGMN on the EBRW-ordered equations, and CGNR/CGNE.

3.3 Supplementary experiments

Fig. 5 shows the results for $r = 100$, i.e., the number of equations is 201. We can see that the relative behavior of the different methods is essentially similar to the case shown in Fig. 3.

Fig. 6 compares the three CG algorithms, namely, CGMN on the EBRW-ordered equations, CGNR and CGNE. The latter two methods were ran on the normalized equations. The results of the ARK method of Liu and Wright [22] are also shown, together with an FG-weighted version of ARK (ARK-FG). The figure also shows the performance of the randomized Cimmino method, with a subset size equal to $\lfloor m/4 \rfloor$.

To compare the number of operations of CGNR and CGNE with the projection methods, we consider the following: each iteration of CGNR and CGNE involves two matrix-vector products, requiring $2mn$ multiplications, and additional inner products requiring $2m + 3n$ multiplications. Also, a single KACZ projection requires $2n$ multiplications, so one cycle of C-KACZ, which consists of m projections, requires $2mn$ multiplications. We therefore equate every iteration of CGNR and CGNE with m projections.

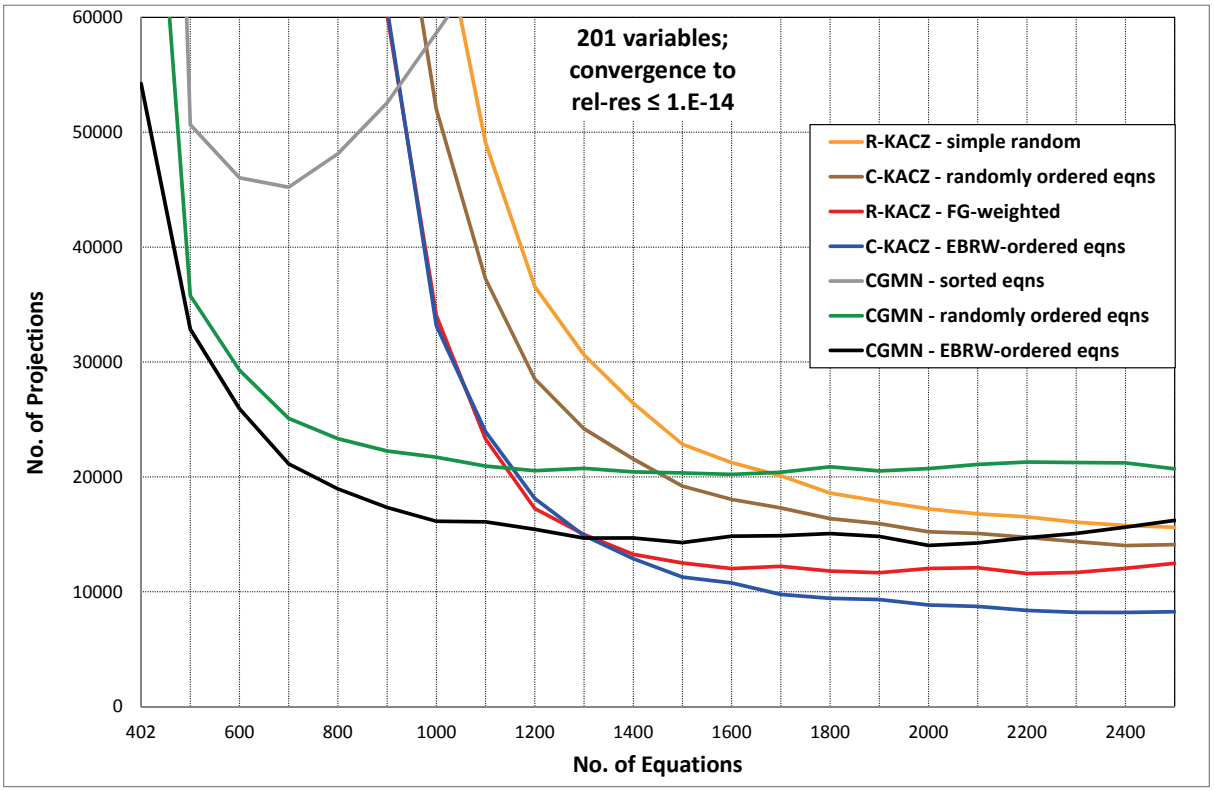

Figure 5: Plot of no. of projections to obtain a relative residual of at least 10^{-14} vs. no. of equations, for 201 variables. C-KACZ is cyclic Kaczmarz, R-KACZ is randomized Kaczmarz.

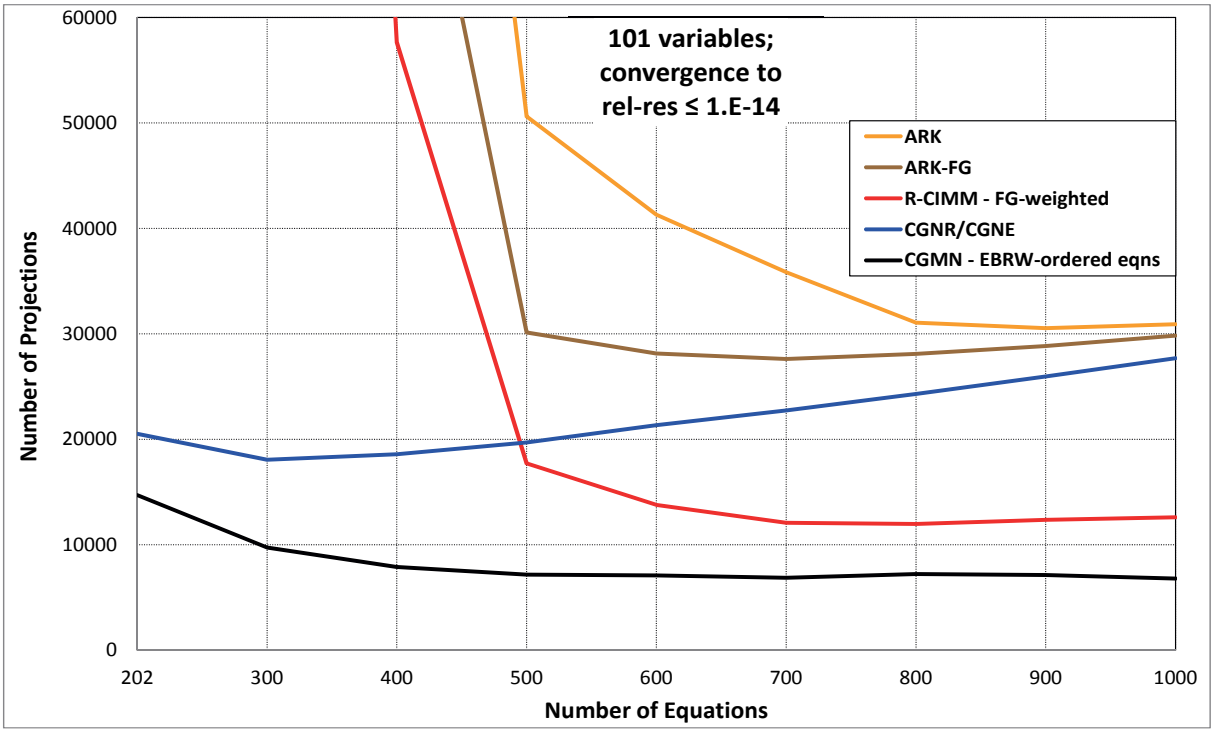

Figure 6: Comparison of the CG methods, ARK [22], an FG-weighted version of ARK, and an FG-randomized version of Cimmino with subset size = $\lfloor m/4 \rfloor$.

As to ARK, we implemented Algorithm 2 from [22]. This requires a choice of $\lambda \in [0, \lambda_{\min}]$, where $\lambda_{\min}$ is the minimal eigenvalue of A. According to [22, §6.3], $\lambda_{\min} \approx (\sqrt{m/n} - 1)^2$. So we set $\lambda = \alpha(\sqrt{m/n} - 1)^2$, where $\alpha < 1$ was a parameter. A few sample runs indicated that $\alpha = 0.2$ produced the optimal results.

In step 7 of ARK, the hyperplane on which a projection is made is chosen with equal probability. ARK-FG modifies this choice and chooses the hyperplane according to the FG-weights. It can be seen that the FG version of ARK is significantly better than the regular version; however, similarly to other results, the significance of the FG-weights diminishes as the number of equations increases.

It is interesting to note that ARK, ARK-FG and R-CIMM-FG exhibit typical Kaczmarz-style plots as in Figures 3 and 5: the required number of projections rises very steeply below 500 equations. The plots of CGNR and CGNE are virtually identical and they increase almost monotonically after 300 equations. This is in keeping with the finding of Strohmer and Vershynin [28] that for 700 equations, R-KACZ-FG is better than CGNR.

We also tried the regular Cimmino algorithm, but the results were too high to even fit in the chart. The minimum number of projections required by CIMM was more than 92,000, obtained at 1,000 equations. The optimal relaxation parameters for CIMM and R-CIMM-FG were 85 and 75, respectively.

4 Additional considerations

This section deals with some additional issues which require further consideration: performance of the algorithms when the maximal gap size is increased, and methods of extending the FG-weights and EBR to higher dimensions.

4.1 Gap size experiments

Feichtinger et al. [12] present important results tying the maximal gap size to the robustness of several reconstruction methods. Ideally, the maximal gap size, which we will denote by δ, should satisfy the inequality $\delta < 1/(2r)$, where r is the bandwidth.

We consider the case with $r = 50$ and $n = 300$, which gives us 101 variables and 300 equations. As can be seen from Fig. 3, CGMN was the only projection-based method that could tackle this case. We applied CGMN to the EBRW-ordered equations, with maximal gap size ranging from 0.01 to 0.12, and set the stopping criterion with relative residual $\leq$ 1.E-10. The results are shown in Table 1. As can be seen, up to a gap size of 0.05, there is no significance difference between the various results. Note that 0.05 is 5 times larger than the condition set by the above

inequality for δ. We can also see that up to a gap size of 0.1, CGMN provides reasonable practical results, though at a heavier price in the number of projections and runtime.

gap	proj	relres	relerr	time (S)
none	7977	2.826E-11	3.977E-10	0.0114
0.01	8172	2.256E-11	3.074E-10	0.0116
0.02	8128	2.664E-11	3.629E-10	0.0115
0.03	10353	2.036E-11	2.666E-10	0.0147
0.04	11671	2.438E-11	3.046E-10	0.0166
0.05	13622	2.025E-11	4.108E-10	0.0194
0.06	18835	3.752E-11	1.989E-08	0.0268
0.07	23486	3.044E-11	4.661E-08	0.0334
0.08	33069	4.745E-11	6.143E-07	0.0471
0.09	52120	5.098E-11	1.896E-05	0.0739
0.10	79401	5.498E-11	8.397E-04	0.1127
0.11	188172	8.388E-11	7.549E-03	0.2671
0.12	496735	1.645E-10	8.630E-02	0.7069

Table 1: No. of projections, relative residual, relative error, and time required by CGMN (on the EBRW-ordered equations) to achieve relative residual $\leq$ 1.E-10, as a function of the maximal gap size. CGMN was limited to 1000 iterations, and each entry in the table is the average of 100 runs.

Another point to note from Table 1 is that up to a gap size of 0.11, the relative residual goal can be achieved, while the relative error steadily deteriorates when $\delta \geq 0.06$. This means that as δ increases, CGMN can continue to solve the system of equations, but the modeling of the problem by the equations deteriorates. As noted by one reviewer, the steep rise in the number of projections below 500 samples (for the non-accelerated Kaczmarz methods) may be due to the increased chance of having big gaps in the samples.

4.2 Extension to higher dimensions

A natural question that arises is how can the concepts of the FG-weights and the EBR ordering be generalized to higher dimensions? With respect to the FG-weights, the answer is simple: assuming that the sample points lie in some bounded domain of the k-dimensional Euclidean space, we can construct the corresponding Voronoi diagram; this is a well-known concept in computational geometry: the domain is divided into disjoint convex cells, each containing a single sample point. The cell containing a sample point p is the set of all points in the domain that are closer to p than to any other sample point. Then, the FG-weight of every sample point p is taken in proportion to the k-dimensional volume of the Voronoi cell containing p. Note that for $k = 1$, we get the FG-weights that were used in [28] and in this paper.

As to the EBR ordering, there can be several approaches. The simplest approach is to consider

the projections of the samples onto one of the coordinate axes, sort the projections according to that coordinate, and then reorder the points according to the EBR permutation of the projections. A somewhat better method, particularly when the samples exhibit some linear correlation, is to construct the straight line L minimizing the sum of the squares of the distances of the sample points to L. We can then sort the projections of the samples on L and take the EBR-ordering of the projections.

A different approach is to use another tool from computational geometry – a variation of the k-d tree, where k is the dimension of the underlying Euclidean space. Assume that the points lie in some k-dimensional rectangular domain D_0, and the axes are labeled $x_1, \ldots, x_k$. Consider the projections of the sample points on all the axes. Our variation of the k-d tree is a binary tree with D_0 as the root, and constructed as follows. Starting from x_1, we partition D_0 into two subdomains, D_1 and D_2, by passing a plane orthogonal to the x_1-axis so that the number of points in D_1 and D_2 differ by at most one. (Contrary to a regular k-d tree, the partitioning plane does *not* pass through a sample point.) D_1 and D_2 are the two sons of D_0, and they are each split into two subdomains by hyperplanes orthogonal to the x_2-axis. This procedure continues by using the axes in cyclic order. A subdomain containing a single sample point becomes a leaf.

An example of this construction for two dimensions is shown in Fig. 7, with the partitioning lines at different levels shown in different colors. Note that if the number of points is not a power of 2, then not all leaves will be at the same level. We now order the samples by the inorder of the leaves of the k-d tree, and then apply the EBR permutation to this order. Fig. 7 shows both orders. It can also be noted that the k-dimensional volume of each leaf can serve as an alternative to the Voronoi cell for the FG-weights, though it is not as accurate as the Voronoi approach for $k > 1$.

5 Conclusions

The problem of recovering bandlimited functions from random samples has been examined from several angles: from relatively few samples to a high number of samples, various Kaczmarz-based algorithms, and a novel derandomization method intended to avoid the problematic issue of adjacent equations whose hyperplanes form a small angle. Similarly to the results of [28], use was made of certain weights which measure the "isolation" of each sample w.r.t. its neighbors.

The derandomization technique is a method of shuffling the equations using an extension of the bit-reversal permutation, and called EBR. The sequence of equations ordered according to EBR is augmented with a certain fraction of the equations with the largest weights. This combined list of equations is called the EBRW-ordered equations. The advantage of using EBR is twofold: it ensures that all the equations will be used, and most importantly, it enables the use of CGMN

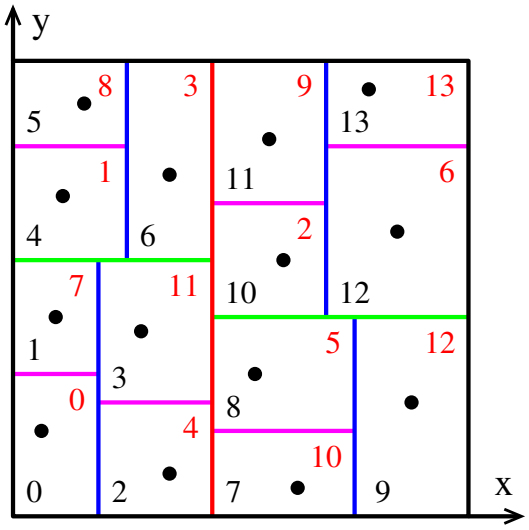

Figure 7: A k-d tree partitioning of 14 points in the plane ($k = 2$, and partitionings are chosen between points). The numbers in black are the regular inorder of the leaves, and the numbers in red show the EBR ordering of the samples.

which accelerates the regular Kaczmarz and solves the problem at low sampling rates.

The main findings of our results is that for low to moderate ratios of the equations to variables, the best results are obtained when CGMN [1] is applied to the EBRW-ordered equations. For moderate to high ratios, the best results are obtained with the original (cyclic) Kaczmarz applied to the EBRW-ordered equations.

When the equations/variables ratio (EVR) falls below 5, all the Kaczmarz variants show a sudden and very steep increase in the number of projections required to reach a given relative residual goal. CGMN, which is a CG acceleration of Kaczmarz, may also exhibit this behavior, but at a much lower value of EVR and to a much lesser extent, particularly when it is applied to the EBRW-ordered equations. It was also found that signals composed mainly of high-frequency components are easier to recover, and this may have some practical applications.

The experiments indicate that sorting the equations according to the order of the samples should be avoided, for all the cyclic Kaczmarz variants, and also for CGMN. This is due to the fact that in such an ordering, every pair of adjacent equations contributes two hyperplanes with a very small angle between their normals. This is a worst-case situation for Kaczmarz, and it was the motivation for introducing the EBR permutation.

Finally, it is shown that the FG-weights and the EBR-ordering can be extended to higher dimensions by using standard tools from computational geometry.

Acknowledgments. The author would like to thank the anonymous reviewers for their helpful comments. §4 was added in response to the issues raised by one of the reviewers.

References

[1] Å. Björck and T. Elfving. Accelerated projection methods for computing pseudoinverse solutions of systems of linear equations. *BIT*, 19:145–163, 1979.

[2] M. Blum, R. W. Floyd, V. Pratt, R. L. Rivest, and R. E. Tarjan. Linear time bounds for median computations. In *Proc. 4th Annual ACM Symp. on Theory of Computing*, STOC '72, pages 119–124, New York, 1972. ACM.

[3] C. Byrne. A unified treatment of some iterative algorithms in signal processing and image reconstruction. *Inverse Problems*, 20:103–120, 2004.

[4] C. Cenker, H. 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 . In P. Maragos, editor, *Visual Communications and Image Processing '92*, pages 299–310. SPIE, Nov. 1992.

[5] Y. Censor, T. Elfving, and G. T. Herman. Averaging strings of sequential iterations for convex feasibility problems. In D. Butnariu, Y. Censor, and S. Reich, editors, *Inherently Parallel Algorithms in Feasibility and Optimization and Their Applications*, volume 8 of *Studies in Computational Mathematics*, pages 101–113. Elsevier, Amsterdam, 2001.

[6] Y. Censor, G. T. Herman, and M. Jiang. A note on the behavior of the randomized Kaczmarz algorithm of Strohmer and Vershynin . *J. of Fourier Analysis and Applications*, 15:431–436, 2009.

[7] G. Cimmino. Calcolo approssimato per le soluzioni dei sistemi di equazioni lineari. *La Ricerca Scientifica XVI, Series II, Anno IX*, 1:326–333, 1938.

[8] J. W. Cooley and J. W. Tukey. An algorithm for the machine calculation of complex Fourier series . *Mathematics of Computation*, 19:297–301, 1967.

[9] J. Demmel. The probability that a numerical analysis problem is difficult. *Mathematics of Computation*, 50(182):449–480, 1988.

[10] Y. C. Eldar and D. Needell. Acceleration of randomized Kaczmarz method via the Johnson-Lindenstrauss lemma . *Numerical Algorithms*, 58:163–177, 2011.

[11] H. Feichtinger and K. Gröchenig. Theory and practice of irregular sampling. In M. Frazier, editor, *Wavelets: Mathematics and Applications*, pages 305–363. CRC Press, Boca Raton, Florida, USA, 1994.

[12] H. G. Feichtinger, K. Gröchenig, and T. Strohmer. Efficient numerical methods in non-uniform sampling theory. *Numerische Mathematik*, 69:423–440, 1995.

[13] D. Gordon and R. Gordon. Component-averaged row projections: A robust, block-parallel scheme for sparse linear systems. *SIAM J. on Scientific Computing*, 27:1092–1117, 2005.

[14] D. Gordon and R. Gordon. CGMN revisited: robust and efficient solution of stiff linear systems derived from elliptic partial differential equations. *ACM Trans. on Mathematical Software*, 35(3): 18:1–18:27, Oct. 2008.

[15] D. Gordon and R. Gordon. Solution methods for linear systems with large off-diagonal elements and discontinuous coefficients. *Computer Modeling in Engineering & Sciences*, 53(1):23–45, Nov. 2009.

[16] D. Gordon and R. Gordon. CARP-CG: a robust and efficient parallel solver for linear systems, applied to strongly convection-dominated PDEs. *Parallel Computing*, 36(9):495–515, Sept. 2010.

[17] D. Gordon, R. Gordon, and E. Turkel. Compact high order schemes with gradient-direction derivatives for absorbing boundary conditions. *J. of Computational Physics*, 297(9):295–315, Sept. 2015.

[18] G. T. Herman. *Fundamentals of Computerized Tomography: Image Reconstruction From Projections*. Springer, 2nd edition, 2009.

[19] G. T. Herman and L. B. Meyer. Algebraic reconstruction techniques can be made computationally efficient. *IEEE Trans. on Medical Imaging*, MI-12:600–609, 1993.

[20] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49:409–436, 1952.

[21] S. Kaczmarz. Angenäherte Auflösung von Systemen linearer Gleichungen. *Bulletin de l'Académie Polonaise des Sciences et Lettres*, A35:355–357, 1937.

[22] J. Liu and S. J. Wright. An accelerated randomized Kaczmarz algorithm . *Mathematics of Computation*, 85(297):153–178, Jan. 2016.

[23] E. Margolis and Y. C. Eldar. Nonuniform sampling of periodic bandlimited signals. *IEEE Trans. on Signal Processing*, 56(7):2728–2745, Jul. 2008.

[24] M. Mayer. *POCS-Methoden*. PhD thesis, University of Vienna, Austria, Oct. 2000. http://univie.ac.at/nuhag-php/bibtex/open_files/ma00_mayerPOCS.pdf.

[25] D. Needell. Randomized Kaczmarz solver for noisy linear systems . *BIT – Numerical Mathematics*, 50:395–403, 2010.

[26] Y. Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM J. on Optimization*, 22(2):341–362, 2012.

[27] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, Philadelphia, PA, 2nd. edition, 2003.

[28] T. Strohmer and R. Vershynin. A randomized Kaczmarz algorithm with exponential convergence . *J. of Fourier Analysis and Applications*, 15:262–278, 2009.

[29] J. A. Tropp, J. N. Laska, M. F. Duarte, J. K. Romberg, and R. G. Baraniuk. Beyond Nyquist: efficient sampling of sparse bandlimited signals . *IEEE Trans. on Information Theory*, 56(1):520–544, Jan. 2009.