ACCELERATING SPARSE RECOVERY BY REDUCING CHATTER*

EMMANOUIL DASKALAKIS †, FELIX J. HERRMANN‡, AND RACHEL KUSKE§

Abstract. Compressive Sensing has driven a resurgence of sparse recovery algorithms with ℓ_1-norm minimization. While these minimizations are relatively well understood for small under-determined, possibly inconsistent systems, their behavior for large over-determined and inconsistent systems has received much less attention. Specifically, we focus on large systems where computational restrictions call for algorithms that use randomized subsets of rows that are touched a limited number of times. In that regime, ℓ_1-norm minimization algorithms exhibit unwanted fluctuations near the desired solution, and the Linear Bregman iterations are no exception. We explain this observed lack of performance in terms of chatter, a well-known phenomena observed in non-smooth dynamical systems, where intermediate solutions wander between different states stifling convergence. By identifying chatter as the culprit, we modify the Bregman iterations with chatter reducing adaptive element-wise step lengths in combination with potential support detection via threshold crossing. We demonstrate the performance of our algorithm on carefully selected stylized examples and a realistic seismic imaging problem involving millions of unknowns and matrix-free matrix-vector products that involve expensive wave-equation solves.

Key words. sparsity promotion, inconsistent linear systems, Kacmarz, linearized Bregman dynamical systems, non-smooth dynamics, chatter

AMS subject classifications. 68Q25, 68R10, 68U05

1. Introduction.

In compressive sensing, a hot field in applied mathematics for the last fifteen years, the key assumption is that the unknown solution vector $x \in \mathbb{R}^n$, is sparse (most of its entries are zero). Given a data vector $b \in \mathbb{R}^m$ and a sensing matrix $A \in \mathbb{R}^{m \times n}$ with $(m \ll n)$ such that $Ax = b$, recovering x is non-trivial because it is an under-determined problem. However, this problem can be solved using the assumed sparsity, under certain conditions on the matrix A, via a ℓ_1-norm minimization procedure [2, 3, 7]. While there are many options for solving ℓ_1- norm minimization problems, we focus on a modification of the linearized Bregman iterations given below (cf. Equation (1.1)), a method well-suited for ℓ_1-norm optimization problems with convergence guarantees [1].

The linearized Bregman (LB) method [26] follows a simple iterative scheme involving

$$\begin{aligned} z_{k+1} &= z_k - t_k A^\top (Ax_k - b) \\ x_{k+1} &= S_\lambda(z_{k+1}), \end{aligned} \tag{1.1}$$

where $S_\lambda(z_k) = \max(|z_k| - \lambda, 0)\,\mathrm{sign}(z_k)$ is a soft thresholding or shrinkage nonlinearity, $A^\top$ is the transpose of A and t_k is the step length or time step (cf. 1.2). These iterations converge in the limit $\lambda \uparrow \infty$ to the solution of the well known Basis Pursuit (BP) problem—i.e.,

$$\min_x \|x\|_1 \quad \text{subject to} \quad Ax = b \tag{BP}$$

*Submitted to the editors October 3, 2019.

Funding: This research was carried out as part of the SINBAD II project with the support of the member organizations of the SINBAD Consortium. Additional partial funding was provided by a NSERC Discovery Grant, a NSERC Discovery Accelerator Supplement, and a Simons Foundation Fellowship for work carried out in part at the Isaac Newton Institute. The authors also thank Georgia Tech for research support.

†Vancouver Community College, Vancouver BC, Canada (edaskalakis@vcc.ca).

‡Seismic Laboratory for Imaging and Modeling (SLIM), Georgia Institute of Technology (felix.herrmann@gatech.edu).

§School of Math, Georgia Institute of Technology (rachel@math.gatech.edu).

where $\|x\|_1$ is the l_1-norm, $\|x\|_1 = \sum_{i=1}^{n} |x^i|$ for x^i the entries in x.

We follow [12] and use "dynamic time steps" t_k's given by

$$t_k = \frac{\|Ax_k - b\|_2^2}{\|A^\top (Ax_k - b)\|_2^2}. \tag{1.2}$$

For consistent (noise-free) systems, the above iterations Equation (1.1) with dynamic time steps t_k converge to the solution of the following strongly convex optimization problem:

$$\min_x \lambda\|x\|_1 + \frac{1}{2}\|x\|_2^2 \quad \text{subject to} \quad Ax = b\,. \tag{EL$_\lambda$}$$

The combination of the ℓ_1- and ℓ_2-norms makes the above problem strictly convex and is referred to as an elastic net in the machine learning literature [27]. By including projections onto a ℓ_2-norm ball of size σ [12] that equals the ℓ_2-norm of the noise, the above iterations can be used to solve inconsistent problems of the type

$$\min_x \|x\|_1 \quad \text{subject to} \quad \|Ax - b\|_2 \le \sigma\,, \tag{BPDN$_\sigma$}$$

known as Basis Pursuit DeNoise (BPDN). Since $Ax = b$ is not solved exactly, this formulation avoids fitting the noise, and thus small coefficients are not captured.

While convergence of Equation (1.1) for consistent ℓ_1-norm minimization problems has been established [1], we are interested in large inconsistent and over-determined problems for which exact recovery is not possible. Since our problems are large and over-determined, we study the behavior of iterations that involve (random) subsets of data—i.e., we follow [25] and write

$$\begin{aligned} z_{k+1} &= z_k - t_k A_k^\top (A_k x_k - b_k) \\ t_k &= \frac{\|A_k x_k - b_k\|_2^2}{\|A_k^\top (A_k x_k - b_k)\|_2^2} \\ x_{k+1} &= S_\lambda(z_{k+1}), \end{aligned} \tag{LB$_\text{k}$}$$

where the pair $\{A_k, b_k\}$ represents (randomly) chosen subsets of rows selected from A and the corresponding data points b_k extracted from b. As a result, we solve a sequence of sub-problems involving sub-matrices $A_k = A_{r(k)}$ that consist of randomly sub-sampled rows $r(k)$ taken from the tall matrix A (see Figure 1) and redrawn with replacement during each kth iteration. Then the iterations correspond to a sparse block-Kaczmarz method that is connected to the linearized Bregman method. When the iterations use a single row at each step, the iterations become Kaczmarz iterations [16, 19] as shown by [12, 13]. These authors also demonstrated for the undetermined problem that, irrespective of the batch size (number of rows in A_k), the iterations in LB$_\text{k}$ converge to the solution of the original problem BP when $\lambda \uparrow \infty$.

Even though the extension in BPDN$_\sigma$ to include noisy problems allows for solutions of inconsistent under-determined systems, its behavior is much less well studied and understood in situations where A is tall and the size of the problem dictates that we can use only a limited number of pairs of $\{A_k, b_k\}$, because these row blocks are expensive to evaluate. That is, we have to work under the condition that *(i)* we can afford only a few passes (epochs) through the data (one pass through the data corresponds to the numbers of iterations necessary to touch m randomly selected rows); *(ii)* the condition number of the matrix may not be good, in contrast to compressive-sensing problems where the matrix is well-conditioned by design; *(iii)* unknown vectors

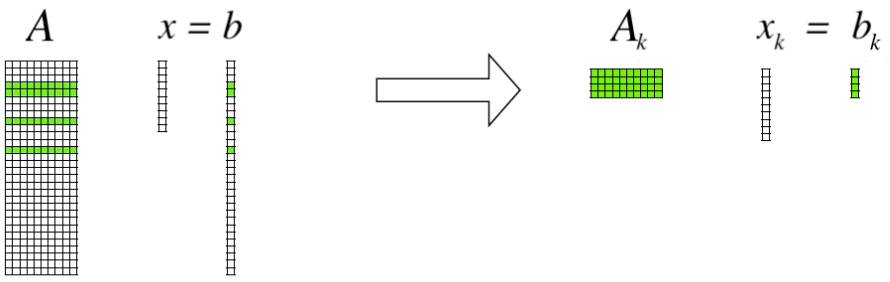


Fig. 1: The subsampling process (figure adapted from [5])

x are compressible rather than strictly sparse, meaning that the size distribution of the entries of x includes small values. Then we can hope to approximate the solution only up to small coefficients remaining in the tail of this distribution. That is, in essence we approximate BP by solving EL_λ with a limited number of iterations for finite λ. With this approach, we look to capture small coefficients without fitting the noise.

Given the focus on large, inconsistent systems, we seek to approximately invert systems of this type by allowing only a limited number of passes through the data and restricting our attention to systems that also behave "compressive sensing like". Specifically, we mean that x is strictly sparse or can be well approximated by a small number of non-zero entries and that under certain conditions [8] the following relationship holds:

$$A_k^\top b_k \approx \alpha I x + n_k,$$

where the A_k's are flat, $\alpha \leq 1$ and n_k Gaussian-like noise. The size of this noise depends on the number of rows in A_k and on the level of inconsistency of the system of equations. Even though the systems of interest are inconsistent, we study the above iterations without including projections onto the ℓ_2 norm ball. We justify this choice by the following arguments: *(i)* we are only allowed to make a very limited number of passes through the data, which limits the risk of fitting the "noise", which is a common concern in BPDN_σ; *(ii)* we know from experience that solutions of BPDN_σ miss the small entries in x that are of interest to us, given the relaxed data-fit constraint; *(iii)* in practice, our "noise" is not incoherent random noise, thus violating the assumptions behind the additional projections. Furthermore, as demonstrated at the end of Section 3, these projections do not address certain fluctuations related to stalling that we address in this paper.

Under these assumptions, one could argue that LB iterations as in Equation (1.1) would make the most progress because well-tuned thresholding should be able to separate the interference noise from spiky signal. Because our matrices of interest are "compressive sensing like" to a limited degree, the above ideal "denoising" scenario of LB does not hold. This shortfall explains, at least in part, why the LB algorithm struggles to converge rapidly to a solution in these cases.

Many attempts have been made in the literature to speed up the convergence of ℓ_1-norm minimization problems. For first order methods—i.e., methods that use

first-order derivative information on the objective only, these range from relaxing the constraints, known as a homotopy [21], to techniques derived from belief propagation or from dynamical systems. The former is known as approximate message passing [8, 11] and relies on a delicate and therefore impractical set of assumptions on the pairs $\{A_k, b_k\}$ [14]. The latter involves the inclusion of an additional memory term as proposed by Nesterov. Unfortunately, we found empirically that none of these approaches are adequate to handle our problems of interest that are large, inconsistent, and mildly ill-conditioned. Furthermore, we note the contrast between the iterative shrinkage thresholding algorithm (ISTA) that updates from the value x_k following from the thresholding operation S_λ, and the LB method Equation (1.1) that updates from the value z_k of the last iterate before thresholding (i.e. for ISTA, x_k replaces z_k on the right hand side of the first equation of Equation (1.1)). This difference allows LB iterations to hone in on sparse solutions more quickly and flexibly than ISTA (as demonstrated at the end of Section 3).

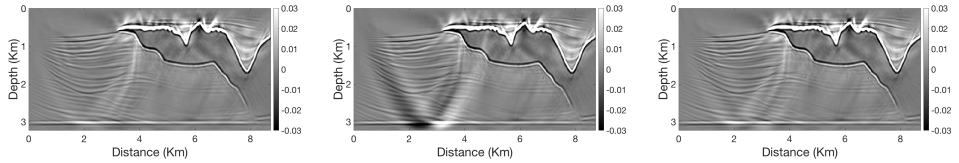
Because of its relative simplicity and connection to the Kaczmarz method [16, 19], we take the LB iterations in Equation (1.1) as a starting point. We empirically study these iterations from the perspective of a nonlinear non-smooth dynamical system with noise. In particular, we associate the observed and reported stalling behavior of Equation (1.1) when applied to inconsistent systems (also observed for other algorithms such as ISTA), with the phenomenon of chatter well-known in non-smooth dynamics [20]. We demonstrate that chatter is responsible for relatively strong fluctuations in the model iterates x_k preventing the smaller entries of x to enter into the solution.

1.1. Organization. After briefly providing a practical motivation for our problem in Section 2, we discuss this phenomenon of chatter in detail by describing a series of carefully selected numerical experiments in Section 3 including a simple counter measure reducing the chatter. In Section 4, we propose and discuss a more sophisticated method to reduce chatter by exploiting additional structure sparsity-promoting problems offer. In Section 5 we compare the behavior of the model error and residuals, while also illustrating the influence of the choice of the threshold parameter λ in problems with either sparse or compressible solutions. We conclude with Section 6, where we study the impact of our chatter reducing algorithm in the practical setting of Section 2.

2. Practical motivation. While there has been significant progress in the design and implementation of ℓ_1-norm minimization problems, first-order methods including the LB method become challenging when x becomes large and the data misfit constraint becomes expensive to evaluate. A good example of such a problem is Sparsity-Promoting Least-Square Reverse-Time Migration (SPLS-RTM). This problem involves the minimization of a separate data misfit constraint for each source experiment, which in itself requires the solution of a large ill-conditioned but invertible system of equations that discretize a partial differential equation (PDE). After linearization, wave-equation based geophysical sparsity-promoting imaging problems take the form

$$(\text{SPLS}_\sigma) \qquad \min_x \|x\|_1 \quad \text{subject to} \quad \sum_{i=1}^{n_s} \|J_i[m_0, q_i]C^* x - b_i\|_2 \leq \sigma.$$

In this expression, the unknown vector x contains complex-valued curvelet transform coefficients of the image; the matrices J_i, $i = 1 \cdots n_s$ with n_s the number of source



(a) Iteration 21 (b) Iteration 22 (c) Iteration 23

Fig. 2: The resulting migration image for three consecutive iterations.

experiments or "shots", correspond to the discretized linearized Born modelling operator for the i^{th} source experiment; the vector m_0 is the background model for the compressional wavespeed; the q_i and b_i are the source wavelet and observed data for the i^{th} shot; and C^* is the conjugate transpose of the Curvelet transform. Solving problem SPLS_σ is challenging for the following reasons: *(i)* the system is inconsistent because the derivation of the Jacobians $J_i(m_0)$ is based on a linearization with respect to the background model (m_0), thus introducing the inconsistency; *(ii)* evaluations of actions with J_i and $J_i^\top$ are expensive because they involve at least two wave-equation (PDE) solves for each source; *(iii)* there are many source experiments—i.e., n_s is large making iterative solutions that involve all n_s source experiments infeasible; *(iv)* the system of equations in SPLS_σ is ill-conditioned due to physical constraints on the acquisition geometry and the frequency content of the sources; and *(v)* earth images are not strictly sparse but compressible and the challenge is to capture as many small curvelet coefficients as possible.

To provide a concrete example of the challenges that arise in this context, we plot imaging results for iterations $k = 21 \cdots 23$ in Figure 2. Even though the LB_k iterations produce a reasonable image after making approximately two passes through the data (i.e., we touched 10% of the shots 20 times), we observe a cyclic behavior where the solutions are hopping between two different solutions. This type of behavior is consistent with chatter and leads to undesired stalling. Unfortunately, this stalling can lead to serious deterioration of the resulting image because we are typically only allowed to make a limited number of passes through the data.

Remark. Working with sub-problems A_k has advantages when the condition number of the sub-matrix A_k is better than the condition number of the full matrix A and when fast matrix multiplies are available for each block of rows [17]. The latter is certainly the case for Problem SPLS_σ and there are also indications that the conditioning of the sub-problems is better. This means we are in the right regime.

Before we study the behavior of tall inconsistent problems of a realistic size and complexity, we first consider the dynamics of small scale problems in order to expose the source of the chatter. We do this by looking at the relative model error as a function of k for strictly sparse problems. This relative error measures how close x_k is to the exact solution x_{exact} and is defined as

$$\mathcal{M}_e(x_k) = \frac{\|x_{\text{exact}} - x_k\|_2}{\|x_{\text{exact}}\|_2}, \qquad \text{where } Ax_{\text{exact}} = b \;\; (\sigma = 0). \tag{2.1}$$

Unfortunately, the model error requires prior knowledge on the exact solution x_{exact}. For this reason, we also consider the normalized residual. This normalized residual is

given by

$$\mathcal{R}(x_k) = \frac{\|Ax_k - b\|_2}{\|b\|_2}. \tag{2.2}$$

Typically it is computationally infeasible to calculate this normalized residual for large matrices A that encode imaging problems with PDEs so we compute the normalized residuals for each sub-problem instead—i.e., we compute

$$\mathcal{R}_k(x_k) = \frac{\|A_k x_k - b_k\|_2}{\|b_k\|_2}. \tag{2.3}$$

3. Dynamics of a strictly sparse toy problem. We consider the widely observed stalling of (block) Kaczmarz-type iterations for inconsistent over-determined systems from the perspective of chatter, a related phenomenon observed in dynamical systems. We note also the role of the thresholding operator S_λ, an essential part of Equation (1.1) that makes this algorithm well-suited for sparse problems, since S_λ encourages the algorithm to pursue larger entries in a sparse vector as the solution. That is, the thresholding removes small entries that appear in the iterations, and one is left with the model iterate x_k at each step. These properties together with the ease in programming make Equation (1.1) (as well as $\mathrm{LB_k}$) an attractive algorithm.

However, while this thresholding facilitates the fast convergence to exact solutions for (consistent) sparse problems, it can also contribute to a phenomenon known as chatter in the context non-smooth dynamical systems. The evolution of such a system is typically described by different sets of dynamical equations in different regions of solution space; that is, either the coefficients or the forms of the governing equations are discontinuous across a switching surface in the solution-space [6, 20, 18]. In Equation (1.1) the threshold λ introduces non-smooth dynamics so that the equation for each entry z^i is this type of system; specifically, for $|x_k^i| < \lambda$, the i^{th} element of $A^T A x_k$ is replaced by 0. Just as in smooth systems, the governing equations describe how the system moves from any non-equlibrium state toward an equilibrium or attracting state. However, the potential for chatter arises when the attracting state for a given set of governing equations is a virtual equilibrium, that is, if it is in a region with a different set of governing equations. Then the system crosses the switching surface but never reaches that virtual equilibrium. Instead, the system crosses the switching surface and evolves according to the governing dynamics for that region of state space. If there is no stable equilibria or attracting state in any of the regions, the system continues to pursue a sequence of virtual equilibria, crossing the switching surface but never reaching a stable state. This repeated crossing of the switching surface is a typical signature of chatter for non-smooth systems, together with the system never reaching an equilibrium or attracting state. Classic examples of chatter appear in control systems; for example, the simple model of a thermostat, operating a heating or cooling system when a threshold is crossed, with repeated crossings of this threshold if the ambient temperature is not in the temperature range for which the system is off. Noise may also drive chatter in non-smooth systems, for example, if the attracting equilibrium is near the switching surface, so that sufficiently large noise may repeatedly drive the system across this surface.

In the problems considered in this paper, the threshold removes certain entries from z_k, thus removing the influence of certain columns of A in the next iteration. While this operation allows the algorithm to seek sparse solutions quickly, it also contributes to one of the two potential sources of chatter that we see below; for

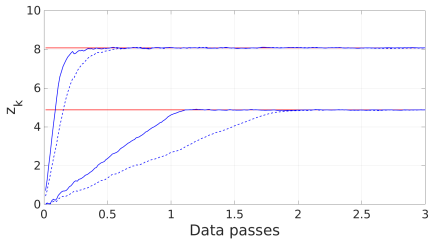
the inconsistent case, there is no attracting solution to EL_λ, corresponding to an equilibrium for Equation (1.1). Then the algorithm continues to seek a solution by moving (small) entries in and out of the support of the solution. Furthermore, the subsampling of A and b at each step can contribute to chatter, since small differences in the behavior of these subproblems yield small differences in the entries of the solution, so those near the threshold can once again move in and out of the support on any given iteration. These different contributions to chatter are discussed further below.

Some theory exists on the behavior of inconsistent systems when operated upon by block Kaczmarz (see [17] equation 1.4), specifically that Kaczmarz iterations stall as soon as the solution gets close to the true solution, continuing to fluctuate about that solution. Their results characterizing this phenomenon depend on the conditioning of the matrix, and are given in terms of a bound on the expectation of $\|x_k - x_{\text{exact}}\|_2^2$, which includes the ℓ_∞-norm of the "noise". Here we are interested in finding an efficient and practical approach to improve performance within the context of LB iterations Equation (1.1) by identifying the above observed phenomena of stalling as chatter. We expect that these results have implications for sparse block Kaczmarz, since the subsampling in the LB iterations reduces to Kaczmarz iterations when taking one row of A at a time.

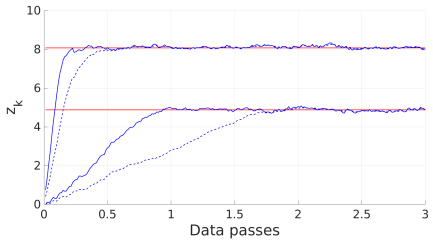
3.1. Drivers of chatter. We investigate different factors contributing to chatter during LB_k iterations by means of a small strictly sparse toy problem. As a first guess, the observed chatter is caused by the interplay between the stochastic dynamics of the LB_k iterations, induced by working with random sub-problems, and the thresholding nonlinearity. To better understand this interplay, we first consider how chatter relates to *(i)* the choice of the time step, *(ii)* the size of the sub-problem $\{A_k, b_k\}$, and *(iii)* whether the problem is consistent ($\sigma = 0$) or not ($\sigma > 0$).

For this small scale demonstration, we first consider problem BP, where the matrix A is a 20000×1000 Gaussian matrix that is drawn from a normal distribution $\mathcal{N}(0, 1)$. To mimic the setting and approach of the motivational example SPLS_σ, we consider the setting where $b = \hat{b} + \varepsilon$, with $Ax_{\text{exact}} = \hat{b}$ and the entries of ε have mean zero and standard deviation σ. Furthermore, we use the iterations LB_k, randomly sub-sampling the data b_k at the kth iteration using a 250×1000 sub-matrix A_k of the matrix A. Since A_k is a random selection of $m_k \ll n$ rows of A redrawn with replacement for each iteration, we work with a different under-determined problem and data at each iteration. We use dynamic time steps [13] and in general λ is determined empirically. In Subsection 5.2, we briefly discuss a possible choice for λ motivated by the dynamics of LB-type algorithms, and leave in-depth discussion to a future paper.

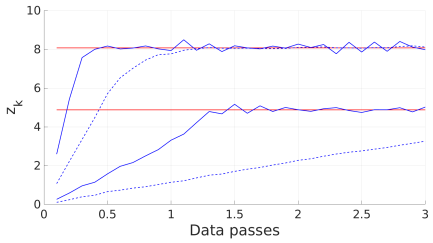
For now, we focus our attention on problems where the vector x_{exact} is strictly sparse, so that the sparsity of the vector x_{exact} is comparable to that of the motivating problem SPLS_σ. To illustrate the chatter, we track the evolution of x_k^i for i corresponding to the non-zero entries $x_{\text{exact}}^i \neq 0$ over three data passes (one data pass or epoch corresponds to 80 iterations when the sub-matrix has 250 rows). As shown in Figure 3, it is insightful to study the dynamics of the system by plotting z_k^i for two fixed indices i that correspond to the largest entry of z_k and one of the small non-zero entries. When the system is consistent $\sigma = 0$, the two entries from z_k^i reach the solution values ($x_{\text{exact}}^i \pm$ the threshold λ) after iterating for approximately $\frac{1}{4}$ of the first data pass (Figure 3a) . In contrast, for the inconsistent case z_k^i reaches the desired values only approximately, at which point the recovered values fluctuate around the desired value, replicating the chattering behavior observed in the motivating example SPLS_σ. To illustrate the effect of the dynamic step length t_k, in Figure 3



(a) $A_k \in \mathbb{R}^{250\times1000}$, $\sigma = 0$



(b) $A_k \in \mathbb{R}^{250\times1000}$, $\sigma > 0$



(c) $A_k \in \mathbb{R}^{2000\times1000}$, $\sigma > 0$

Fig. 3: Plots of two entries of z_k with fixed indices i that correspond to the largest entry and one of the small non-zero entry for the strictly sparse problem. With dashed lines we track entries of z_k that use the constant time step $t_k = \frac{1}{\|A_k\|_2^2}$ and with solid lines we track entries of z_k that use the dynamic time step $t_k = \frac{\|A_k x_k - b_k\|_2^2}{\|A_k^\top (A_k x_k - b_k)\|_2^2}$. The plot in Figure 3a corresponds to the consistent case ($\sigma = 0$) for which no chatter behaviour is observed. On the other hand, plots in Figure 3b and Figure 3c correspond to the inconsistent case ($\sigma > 0$), which suffers from chatter. The red lines indicate the corresponding values of $x^i_{\text{exact}} \pm \lambda$.

we compare the solution trajectories for the inconsistent case using both constant time steps based on the spectral norm, $t_k = \frac{1}{\|A_k\|_2^2}$, and dynamic time steps as in Equation (LB_k), $t_k = \frac{\|A_k x_k - b_k\|_2^2}{\|A_k^\top (A_k x_k - b_k)\|_2^2}$. The latter exhibits faster convergence to the approximate values as expected, but both choices suffer from chatter. Note that the terms constant and dynamic refer to the dependence of t_k on the solution x_k; both constant and dynamic t_k vary with k due to the sub-sample matrix A_k, but only the dynamic time step varies with x_k.

To study the effect of the sub-problems themselves, in Figure 3c we plot trajecto-

ries for both entries when the pairs $\{A_k, b_k\}$ correspond to over-determined systems, where the sub-matrix A_k is tall. For the inconsistent case, the chatter is worse when the pairs of $\{A_k, b_k\}$ correspond to over-determined sub-systems, relative to Figure 3b where the pairs of $\{A_k, b_k\}$ correspond to under-determined sub-systems. While this observation is consistent with reported behavior of stochastic gradient descent [9, 23], where step lengths and line-searches are known to be problematic, the increased chatter for the overdetermined case seems counter-intuitive. It can be explained from the fact that the condition numbers of the submatrices are smaller, an observation originally made by [17]. With this observation, a constant time-step may seem a better choice; yet it is important to remember that the rate of convergence for the constant time step is slower, which is an important consideration for large problems where fewer iterations are desired or necessary. Additional chatter dynamics are discussed in the context of Figure 5 below.

3.2. Controlling chatter with weighted increments. From the small scale experiments above using $\mathrm{LB_k}$ to find x, we conclude that the main contributor to chattering is the inconsistency, combined with the thresholding S_λ. Moreover, choices for the time step t_k and the size of A_k, can make the chatter worse. Since chatter slows down convergence, or even prohibits convergence, we propose a weighted increment approach where we adaptively shorten the time steps t_k, thus reducing the errors resulting from chatter. As reported by [4, 9, 15], convergence of stochastic gradient descent for fixed batch size, i.e., fixed size of the sub-problems, can be realized via a reduction of the overall time step. Tailored to the special structure of our problem, which involves a linear constraint and a strongly convex nonlinear objective designed to exploit sparsity, we design a scheme that reduces the step-size adaptively for each entry depending on its history.

To motivate our adaptive reduction of the time steps, we compare in Figure 4 the actual values of the constant and dynamic time steps selected for the subsampled systems for both the consistent ($\sigma = 0$) and inconsistent ($\sigma > 0$) cases. For the consistent case, the time steps fluctuate but remain centered around a constant value. As expected, the dynamic step lengths are larger and the fluctuations in both cases are due to the random selection of new sub-problem pairs $\{A_k, b_k\}$ for each iteration. However, the situation is completely different for the inconsistent case where the fluctuations in t_k increase with chatter (cf. Figure 3 and Figure 4). As a result, the recovered values for the non-zero entries change from iteration to iteration, in part due to the relatively strong fluctuations and large values of the dynamic time steps at the later iterations.

Because large time steps in the earlier iterations typically lead to preferred faster convergence, we would like to devise an adaptive scheme that allows the solution to make rapid progress in the beginning, followed by a period of increased caution when the chatter sets in. As in stochastic gradient descent [15], such a scheme would reduce the time steps when the stochastic fluctuations start to dominate. Because the chatter differs from entry to entry (see the wiggle trace plot Figure 5, showing the amplitude versus time as an oscillating line), we select the time steps for each entry depending on its history, specifically, the amount of chatter for that entry. To effectively counter the chatter, the scheme must adaptively reduce the chatter more aggressively for those entries with strong chatter. We can accomplish this by keeping track of the signs of the previous gradients. If the signs of the ith entry of these gradients is persistent, having the same sign from iteration to iteration, we want to keep the time step as is. Conversely, we want the time step to decrease when the gradients at the entry changes

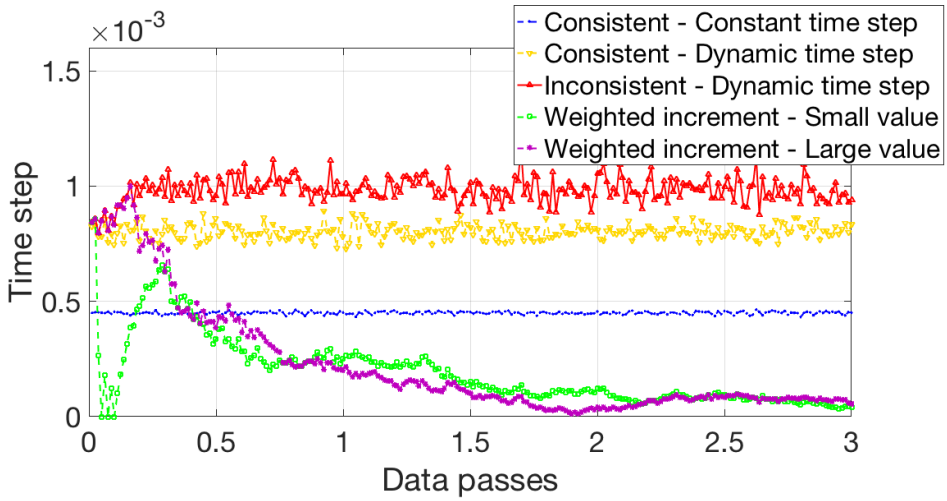


Fig. 4: Here we solve the small scale problem with LB-type approaches, using LB_k for blue, red, and yellow, and tracking the value of the time-step. In blue, we have the constant time step $t_k = \frac{1}{\|A_k\|_2^2}$. In red we have the dynamic time-step $t_k = \frac{\|A_k x_k - b_k\|_2^2}{\|A_k^\top (A_k x_k - b_k)\|_2^2}$ when trying to solve the inconsistent problem. In yellow we have the dynamic time-step when trying to solve the consistent problem with LB_k. With purple and green we track the weighted increments in MLB for the same entries as in Figure 3 (purple is the largest entry and green is a small entry)).

sign often, indicating the onset of chatter. We accomplish these goals by shrinking the ith entry of the time step accordingly, thus assigning a different time step for each entry as

$$\tau_k^i = t_k \frac{\left| \sum_{j=1}^{k} \text{sign}([A_j^\top (A_j x_j - b_j)]^i) \right|}{k}, \quad i = 1 \dots n. \tag{3.1}$$

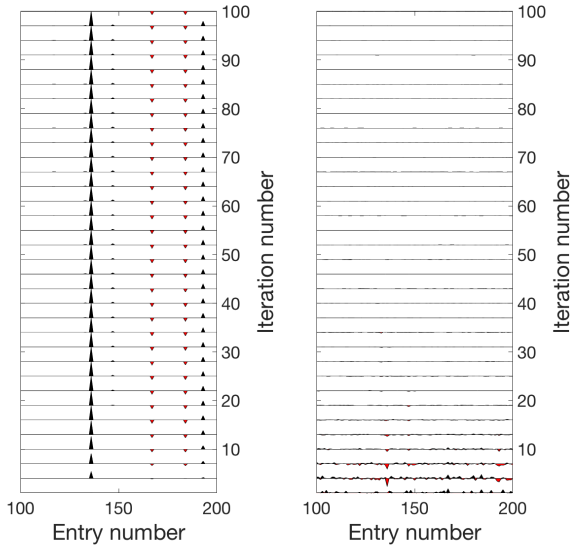
We normalize this expression by the current number of iterations k so that the weights shrink each entry towards zero when chatter sets in—i.e, the element-wise multiplication factor is guaranteed to be between zero and one.

With these weighted increments, our modified Linearized Bregman (MLB) algorithm takes the form:

$$\begin{aligned} z_{k+1} &= z_k - \tau_k \odot A_k^\top (A_k x_k - b_k) \\ x_{k+1} &= S_\lambda(z_{k+1}), \end{aligned} \tag{MLB}$$

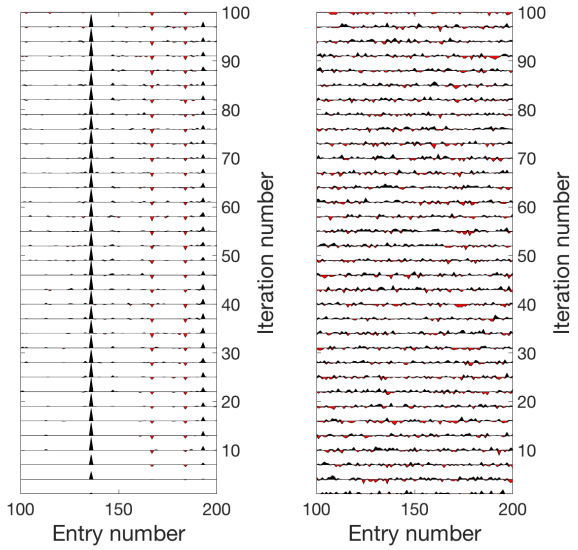
where t_k is either the constant or the dynamic time-step. The symbol $\odot$ denotes the Hadamard product - i.e. element-wise multiplication.

The rationale behind the above choice for the weights in Equation (3.1) is derived from the following three key observations: *(i)* it typically takes ℓ_1-norm minimization relatively few iterations to recover the largest entries of x; *(ii)* often it is not worthwhile to continue iterating once the solution has captured these large entries because of the



(a) Model iterates for $A_k \in \mathbb{R}^{250\times 1000}$, $\sigma = 0$

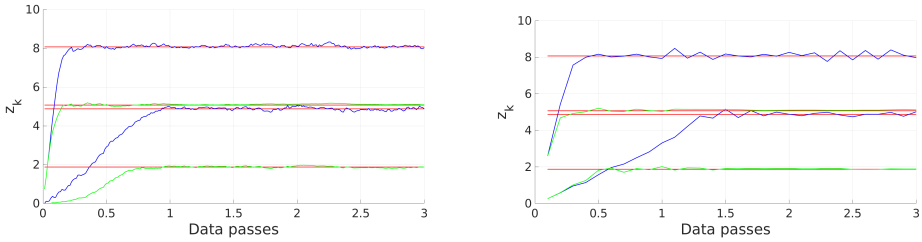
(b) Gradients for $A_k \in \mathbb{R}^{250\times 1000}$, $\sigma = 0$



(c) Model iterates for $A_k \in \mathbb{R}^{250\times 1000}$, $\sigma > 0$

(d) Gradients for $A_k \in \mathbb{R}^{250\times 1000}$, $\sigma > 0$

Fig. 5: Wiggle plots of a part of the model iterates (x_k) and gradients. In the consistent case ($\sigma = 0$), we observe no chatter and the gradient gets smaller as the iterations pass. In the inconsistent case ($\sigma > 0$), we observe the chatter behavior where entries regularly and continually enter and leave the support of the solution. The chatter is driven by the fact that the gradient does not get smaller with the iterations, in contrast to the consistent case.

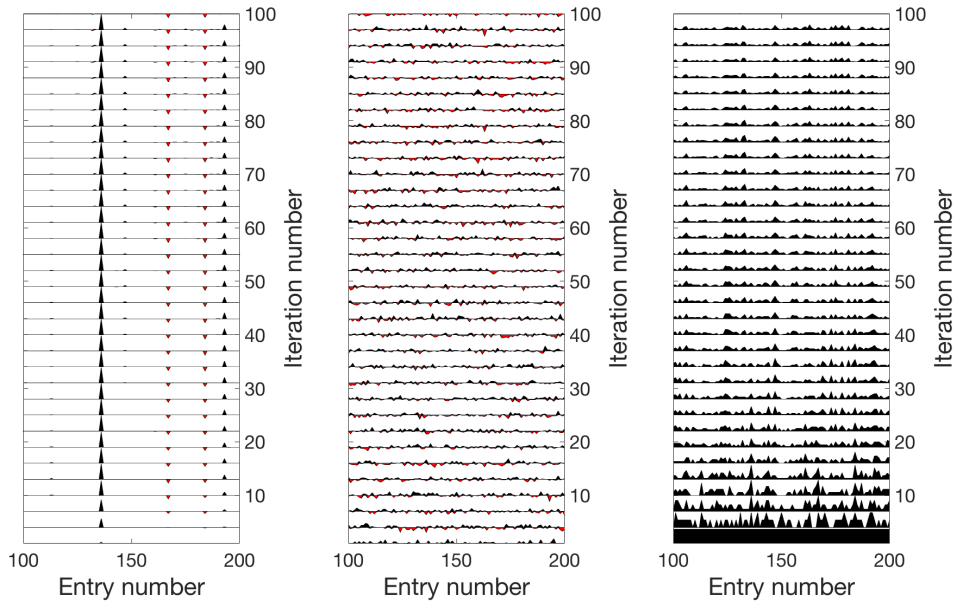


(a) Dynamic time-step, $A_k \in \mathbb{R}^{250\times1000}$ (b) Dynamic time-step, $A_k \in \mathbb{R}^{2000\times1000}$

Fig. 6: The same entries are tracked as in Figure 3 . For entries in green we used the weighted increment τ_k in MLB in comparison to the entries in blue where we use LB_k. The difference in the values of z_k is due to a different choice of threshold for the MLB. Typically $\lambda_{MLB} < \lambda_{LB\text{k}}$. The red lines indicate the corresponding values of $x^i_{\text{exact}} \pm \lambda$, with the values x^i_{exact} the same for both LB_k and MLB.

large-amplitude chatter that develops at these large entries; and *(iii)* it is impossible to recover the many smaller entries as long as the iterations do not settle on the correct values for the large entries. For these reasons, we want to avoid using a large step t_k for an entry that is relatively close to the correct value. This behavior is indeed captured by MLB, with the size of the time steps reduced if the gradients change sign from iteration to iteration — a tell tale sign of chatter. Conversely, the time steps remain large if the sign of the entries in the gradients remain persistent over the iterations.

As shown in Figure 4, we observe exactly this behavior for the large and the small coefficients, where the time steps for both types decrease towards zero as the number of iterations becomes large. However, at the early iterations the weights behave quite differently for large and small coefficients for the reasons mentioned above, namely the sparse recovery algorithm finds the large coefficients first. This explains the small increase in time step early on for these entries, followed by a gradual decrease in the time steps as the values of the large entries are recovered. The small entries, on the other hand, exhibit a different behavior early on. There the fluctuations occurring during the early iterations, while the algorithm seeks the larger values, result in a rapid decay of the time steps for the smaller entries. A rapid increase in the time step follows, as soon as the algorithm starts to hone in on the smaller entries (cf. the green and purple plots in Figure 4). Once the algorithm approaches the desired values for the small entries, the time step decreases as the sustained chatter sets in for these entries. Note that we observe these effects only for inconsistent systems where chatter is expected to occur. By comparing Figure 3 and Figure 6, we observe that our weighting scheme damps the chatter as intended, by shrinking the time steps for the entries where the chatter occurs as shown in Figure 7. There we see that while there are still fluctuations in the gradients, the gradual reduction in time steps damps the chatter. However, this is accomplished at the expense of the rate at which the algorithm reaches the desired values for the smaller entries (cf.Figure 3 and Figure 6). This observation motivates the results of the next section, where an improved weighting scheme is obtained for the smaller entries. This improvement is important especially for compressible signals where the small entries down in the tail are the most difficult to recover.



(a) Model iterates for $A_k \in \mathbb{R}^{250\times1000}$, $\sigma > 0$ (b) Gradients for $A_k \in \mathbb{R}^{250\times1000}$, $\sigma > 0$ (c) Time steps for $A_k \in \mathbb{R}^{250\times1000}$, $\sigma > 0$

Fig. 7: Wiggle plots of a part of the solution, the gradient and the time step using MLB. Even though the gradient has similar behavior compared to the gradient of the LB method (cf.Figure 5d), the chatter is damped as the time step shrinks for certain elements as seen in panel Figure 7c

Before deriving a scheme to accelerate the recovery of compressible signals, let us first briefly compare the solution paths using MLB with those using $\mathrm{LB_k}$-based methods, with and without the projections onto a ℓ_2-norm ball of size σ as defined in [12] and discussed in the Introduction in the context of BPDN_σ. In Figure 8 these solution paths are contrasted also with the iterative shrinkage thresholding algorithm (ISTA) by plotting the ℓ_1-norm against the ℓ_2-norm as the iterative algorithm progresses. For reference, we compare these curves with the Pareto curve [10, 22], which traces the minimal attainable ℓ_2-norm of the residual against the size of the ℓ_1-norm constraint. The strong convexity contributes to the fact that the solution paths for $\mathrm{LB_k}$ stay close to the optimal Pareto curve, in contrast to the solution path of ISTA. Because we invert an inconsistent over-determined system with randomized sampling, the solution paths shown are dominated by random fluctuations when approaching the minimal ℓ_2-norm for the residual. While $\mathrm{LB_k}$ with and without projections on the ℓ_2-norm ball overshoots the true ℓ_1-norm, the MLB method (the green curve) stays close to the Pareto curve and does not overfit the noise, an important feature of our method on which we build below. It is important to note that for $\sigma = 0$ (not shown in the figures), the sustained fluctuations in $\mathrm{LB_k}$ with and without the ℓ_2-norm ball projection are not observed, indicating that they are due to chatter in the inconsistent system, and not due solely to the sampling of sub-problems with replacement.

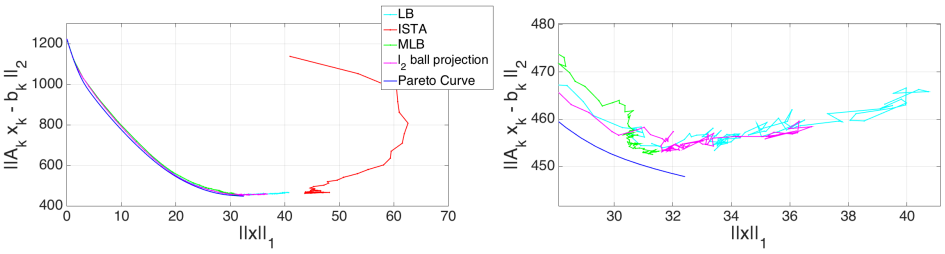
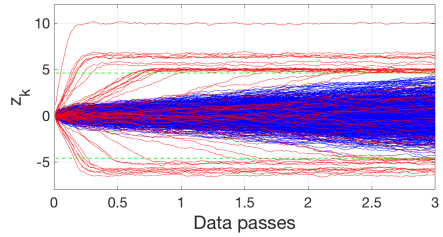


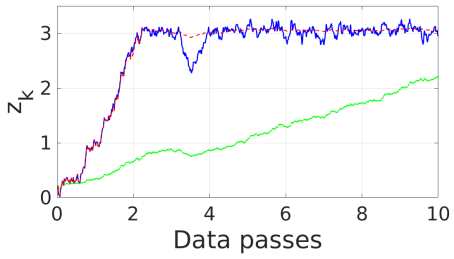
Fig. 8: Solution paths for inconsistent problems, with the right panel a zoom-in of the results shown on the left. The Pareto curve (in blue) is the the optimal solution path for this problem. The solution using $\mathrm{LB_k}$ is plotted with light blue, green is the solution using MLB, ISTA is with red and magenta for $\mathrm{LB_k}$ with the l_2 -ball projection.

4. Acceleration of the modified algorithm. While the proposed weighting scheme MLB addresses some of the issues related to chatter, it can lead to slower convergence especially for the smaller entries. The smaller entries are important for practical problems where the unknown vectors are not sparse but compressible— i.e. both large and small entries are important in the approximation of the solution.

First we re-examine the iteration dynamics of the entries of z_k for the small sparse problem considered in Section 3, in order to analyse the evolution of the update before we apply the threshold at λ. We divide the entries of z_k into two groups, denoted as the large and small solution entries of z_k. The indices of these two groups correspond respectively to the nonzero and zero entries of the exact sparse solution x_{exact}. For consistent problems (not shown), the large entries of z_k reach the desired values for fewer iterations (small k), while the small entries of z_k converge as expected to values below the threshold λ, typically after more iterations (larger k). In contrast, for the entries from inconsistent problems solved with $\mathrm{LB_k}$ as shown in Figure 9a, the inconsistency gives rise to chatter both for the small (blue lines) and large (red lines) entries of z_k, leading to intermittent incorrect support (i.e., the nonzero entries) detection and undesired slow convergence, if at all, as discussed in Section 3 above. While using Equation (3.1) as in MLB significantly reduces the chatter (cf. Figure 6), Figure 9b illustrates the slower convergence of small entries for MLB. There, an entry z^i, with i corresponding to a small contribution x^i_{exact} in the sparse solution, takes a larger number of iterations in MLB to reach the threshold than it does using $\mathrm{LB_k}$. Thus more iterations in MLB are required to capture all contributions to the solution. The slower convergence of the smaller entries in MLB is mainly driven by their initial drop in step size τ^i_k, (see green line in Figure 4), due to strong fluctuations as the algorithm seeks larger entries in the earlier iterations, in turn reducing the step size for smaller entries through τ^i_k in Equation (3.1). While shown below for the sparse case in Figure 9 and Figure 10, obtaining the small entries is particularly essential when the solution is compressible, since detection of the correct support is dependent on capturing the smaller entries. For large scale problems where we can afford only a limited number of passes through the data, a slowdown in capturing the smaller entries can severely limit the performance of MLB.



(a)



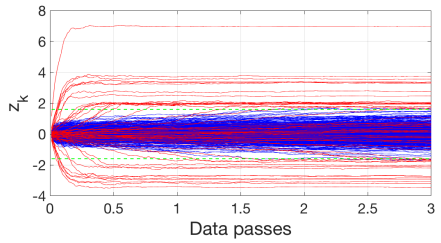
(b)

Fig. 9: (a) Evolution of the entries of z_k, using LB_k for the small scale problem and A_k is 250×1000. With blue we plot the entries z_k^i that correspond to zero valued entries of x_{exact} and with red we plot z_k^i corresponding to the non-zero entries of x_{exact}. (b) Comparison of the evolution of an entry of z_k^i for which the corresponding entry of x_{exact}^i is small in comparison to the maximum of x_{exact}^i, using methods LB_k (blue), MLB (green), and MLB_λ (red). The same threshold value $\lambda = 3$ is used for all methods.

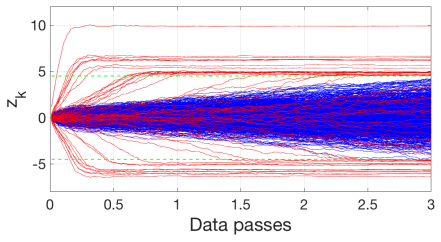
This observation motivates avoiding a small step size in early iterations for the small entries, which we accomplish by including an additional threshold nonlinearity to detect when entries first cross the threshold and potentially enter into the solution. That is, we use the weighted increment τ_k^i only after the first iteration k at which $|z_k^i| > \lambda$. Tracking this threshold crossing corresponds to automatic potential support detection for the weighted increments. In contrast to attempts to detect the support in sparsity-promoting solvers automatically [24], our "support detection" changes only the weighted increments. The resulting update of MLB that tracks threshold crossing is obtained by replacing τ_k with

$$(\text{MLB}_\lambda) \qquad \tau_k^i = \begin{cases} t_k & \text{if } |z_j^i| \le \lambda, \ \ \forall j \le k \\ t_k \dfrac{\left| \sum_{j=1}^{k} \text{sign}([A_j^\top (A_j x_j - b_j)]^i) \right|}{k} & \text{otherwise.} \end{cases}$$

Figure 9b illustrates that the extra element-wise nonlinear operation in MLB_λ can lead to faster convergence when the signal is detectable relative to the noise, particularly in the case where a similar threshold value for λ is used for both LB_k and MLB. MLB_λ avoids choosing a smaller step size prematurely for small entries, so that the behavior of entries below threshold in MLB_λ is the same as LB_k.



(a)



(b)

Fig. 10: Evolution of the entries of z_k for the sparse small scale problem solved with MLB (Figure 10a) and MLB$_\lambda$ (Figure 10b). The threshold parameter λ is represented by the dashed green line.

Figure 10a and Figure 10b show the evolution of small and large entries of z_k for MLB and MLB$_\lambda$, respectively, similar to those shown in Figure 9a. By definition, MLB and MLB$_\lambda$ differ in only the smaller entries of z_k. These entries correspond to the small components of the exact solution, which tend to remain below the threshold longer in MLB even for large k. In Figure 10b, we can see that after two data passes, only a small number of small entries are still below the threshold value for MLB$_\lambda$, while in Figure 10a a smaller threshold is necessary to get comparable results for MLB. In all cases, entries that are not in the support—i.e., corresponding to the zero entries of the exact solution – may occasionally cross the threshold and enter into the solution. For smaller λ, this threshold crossing obviously occurs more frequently within a finite number of iterations, and can lead to over-fitting of the noise as discussed in the next section.

5. Behavior of model error and residuals for compressible problems. We evaluate the performance of each method by comparing using the model error $\mathcal{M}_e(x_k)$ (cf. Equation (2.1)) of the actual solutions obtained from the LB$_\text{k}$, MLB and MLB$_\lambda$ methods. However, in practice the model error is not available, so that one would typically use normalized residuals $\mathcal{R}(x_k)$ (Equation (2.2)) and $\mathcal{R}_k(x_k)$ (Equation (2.3)) instead, and we compare these below. We again consider the inconsistent case, for which it is particularly challenging to resolve compressible solutions, as motivated by the example in Section 2.

5.1. Comparison of model error and residuals. Figure 11 juxtaposes the model errors $\mathcal{M}_e$ for LB$_\text{k}$, MLB and MLB$_\lambda$ for a compressible problem where the solution consists of entries that decay to zero. These figures illustrate how chatter in the LB$_\text{k}$ algorithm results in a plateau for the model error, while removal of the

chatter for MLB and MLB_λ results in a lower model error dependent only on the noise level σ. In Figure 11a, we use the same $\lambda = 3$ for all methods. This value is between the different values of λ tailored to each method that are used in Figure 11b and Figure 11c. While the choice of λ is not the focus of this paper, we see some of its effects in Figure 11 and discuss it briefly in Subsection 5.2 below.

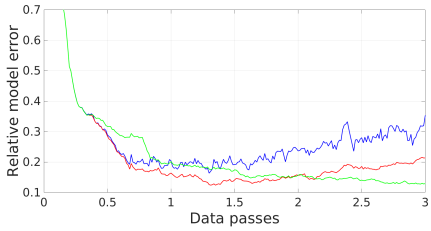
When the same threshold λ is used for all methods, $\mathcal{M}_e(x_k)$ decreases faster for LB_k and MLB_λ during earlier iterations, while MLB takes a larger number of data passes for the $\mathcal{M}_e(x_k)$ to reach the corresponding model error of MLB_λ. The slower reduction in $\mathcal{M}_e(x_k)$ for MLB is directly related to the smaller step size for MLB and slower rate of potential support detection via threshold crossing illustrated in Figure 9b. When a reduced threshold is used for MLB, that method shows a faster decrease of the model error relative to the model error for MLB_λ and LB_k. Furthermore, the results for LB_k and MLB_λ in Figure 11a, illustrate the increase in $\mathcal{M}_e(x_k)$ as k increases, due to over-fitting the noise. For those methods, the value of λ is smaller than the value used in Figure 11b and Figure 11c. This behavior illustrates the danger of overfitting the noise within fewer iterations for smaller values of λ, as mentioned above.

Figure 11c shows results when the matrix A is ill-conditioned. There the chatter drives significantly higher model error for LB_k in comparison to MLB and MLB_λ, both of which reduce the chatter and suffer only from errors related to the magnitude of the noise. The ill-conditioned matrix A is produced by setting the singular values of a Gaussian matrix to give a desired condition number, in this case on the order of 1000.

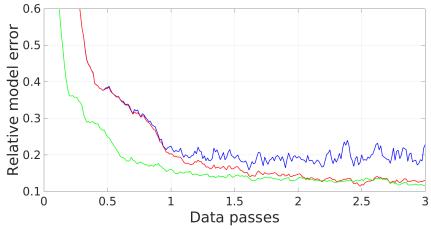
While the ℓ_2-norm relative model error $\mathcal{M}_e$ (Equation (2.1)) can be a useful quantity to compare the performance of the different LB-type algorithms, it requires the oracle of knowing the true solution, which is not available in practice. Because the residual $\mathcal{R}$ (cf. Equation (2.2)) depends on the given matrix A and the data vector b and does not require knowledge of the solution x_{exact}, it is the quantity most often used in practice to compare different iterative methods, rather than the model error. Unfortunately in large scale problems and online compressive sensing [13], often we do not have access to the the full matrix A and data b, or the computation of $\mathcal{R}$ (Equation (2.2)) is very expensive, so the alternative quantity that is typically used is the residual $\mathcal{R}_k$ (Equation (2.3)) using the computationally cheap sub-matrix A_k and data b_k (Equation (2.3)).

In Figure 12, we compare both the residual $\mathcal{R}$ using the full matrix A and $\mathcal{R}_k$ based on the subsampled A_k (Equation (2.2) and Equation (2.3), respectively), for the LB_k, MLB, MLB_λ methods whilst λ is tailored to each. We observe that the computationally inexpensive $\mathcal{R}_k$ does not capture the behavior of the model error (Figure 11b), but $\mathcal{R}$ does. For the purposes of this paper we choose to use the relative model error $\mathcal{M}_e$ for comparison, even though it is not available in real world problems, because it provides an accurate and consistent way to compare different methods, and does not suffer from the subsampling errors of $\mathcal{R}_k$.

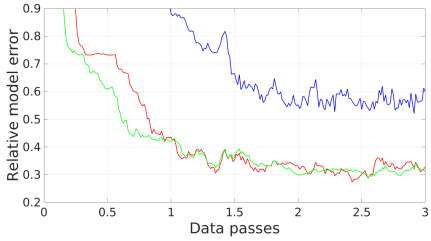
5.2. Influence of the threshold parameter. As seen in Figure 9b and Figure 11, the choice of λ plays a role in the rate at which contributions to the solution are identified, and thus also in the rate at which the model error decreases. On the one hand, when seeking sparse solutions using a LB-type method, a larger threshold value can be advantageous for honing in on larger values over fewer iterations, related to the fact that the iterations in Equation (LB_k) converge to the solution of the original BP when $\lambda \uparrow \infty$ ([13]). On the other hand, taking larger λ can slow the convergence



(a) Compressible solution $\lambda = 2.6$ with $\lambda = 2.6$ used for all methods



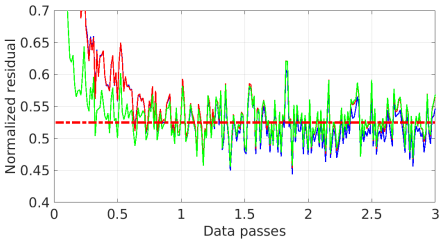
(b) Compressible solution with tailored λ for each method based on Equation (5.1).



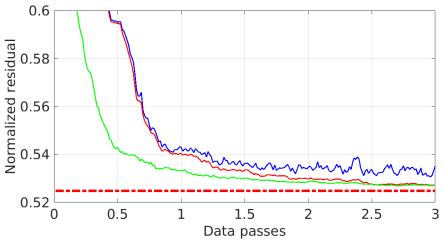
(c) Compressible solution for A an ill-conditioned matrix, using tailored λ based on Equation (5.1).

Fig. 11: Compressible case: The model error $\mathcal{M}_e$ for LB_k with the dynamic time-step (in blue), the MLB with the weighted increment of Equation (3.1) (in green) and the MLB_λ weighted increment of Equation (MLB_λ) (in red).For all panels a)-c), the normalized error is $\frac{\|Ax_{\text{exact}} - b\|_2}{||b||_2} = 0.5248$, providing an approximation to the normalized magnitude of the noise.

for compressible solutions, since a large number of iterations are required for smaller entries to cross the threshold and potentially enter the support. Figure 13b illustrates the dependence of model error on λ when only a limited number of data passes is available. For larger λ, the model error in the compressible case increases with λ, since the smaller entries have not entered the solution. The model error also increases with small, decreasing λ, since the algorithm does not quickly hone in on a compressed sensing-like solution. Furthermore, for a finite number of iterations, a smaller value of λ allows more opportunities to over-fit the noise by admitting incorrect small contributions into the support, thus increasing the error. As shown in Figure 13b, the optimal values of λ for the LB-type methods shown are all $O(1)$ values, above that of



(a) The residual $\mathcal{R}$ of Equation (2.3) for the LB_k (in blue), the MLB (in green) and MLB_λ (in red).



(b) The residual $\mathcal{R}_k$ of Equation (2.2) for the LB_k (in blue), the MLB (in green) and MLB_λ (in red).

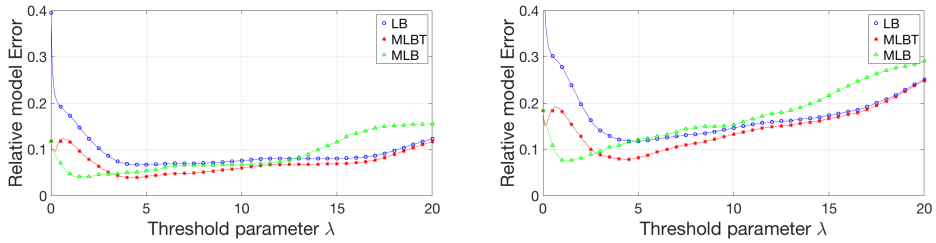
Fig. 12: The two practical ways to calculate the residual (above) using $\mathcal{R}$ (Equation (2.2)) or $\mathcal{R}_k$ (Equation (2.3)). By comparing with Figure 11b, we can see that the behavior only of the residual $\mathcal{R}$ actually resembles the behavior of the model error. The dashed line on both graphs represents the approximation to the normalized noise magnitude $\dfrac{\|Ax_{\text{exact}} - b\|_2}{||b||_2} = 0.5248$.

the noise level.

In Figure 11b, Figure 11c, and Figure 12 we have used an approximation to this optimal value of λ, given by

$$\lambda_{N_d} = N_d \max_{i \in \mathcal{S}}(|[\Delta_\ell G_\ell]^i|), \tag{5.1}$$

where N_d is the number of iterations required for the desired number of data passes and the set $\mathcal{S}$ is $\mathcal{S} = \{i : |z_\ell^i| < \sigma\}$. The quantity Δ_ℓ corresponds to the dynamic times step t_ℓ for LB_k or τ_ℓ^i in MLB or MLB_λ. The term G_ℓ captures the behavior of the gradient, and can be approximated with the average gradient over the first ℓ iterations. The quantities $\Delta_\ell G_\ell$ and z_ℓ are obtained at step ℓ after a number of iterations sufficient for the LB-type methods to hone in on the larger contributions, (e.g. we found ℓ corresponds to approximately 10-20% of a data pass). Intuitively, with this choice of λ, the LB-type method allows enough of the smaller entries of z to cross λ_{N_d} within the desired N_d iterations so that these entries are potentially included in the solution. Simultaneously, the method also avoids over-fitting the noise, since the smallest entries of z corresponding to $x^i_{\text{exact}} = 0$ are unlikely to cross the threshold λ_{N_d}. These properties follow from the fact that for LB-type methods the solution rapidly approaches the larger contributions over the initial iterations, so that at step ℓ the set $\mathcal{S}$ corresponds to the smallest (or zero) entries in x_{exact}. Then $[\Delta_\ell G_\ell]^i$ for $i \in \mathcal{S}$ corresponds to the increments of the (small) entries $z_\ell^i < \sigma$ after the first ℓ



(a) Sparse vector recovery, 3 data passes, $\frac{\|Ax-b\|_2}{||b||_2} = 0.5248$

(b) Compressible vector recovery, 3 data passes, $\frac{\|Ax-b\|_2}{||b||_2} = 0.5248$

Fig. 13: The model error vs the threshold parameter λ . In blue we see results using the original LB_k method, in green are results for MLB and in red are results for MLB_λ. MLB requires a smaller λ to get the minimum model error in comparison to both LB_k and MLB_λ.

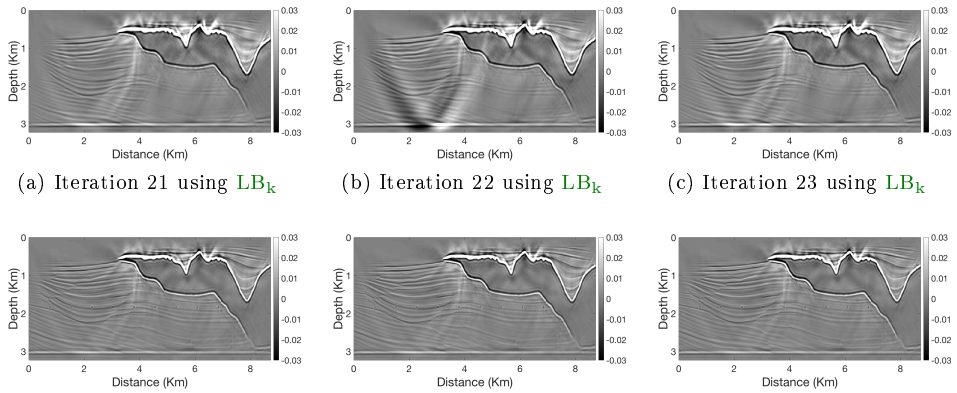
iterations, which require approximately N_d iterations to reach the threshold λ_{N_d}.

Given that Δ_ℓ depends on the method, λ_{N_d} is then smaller for the MLB algorithm, as compared with λ_{N_d} for LB_k and MLB_λ, consistent with the locations of the minima shown in Figure 13. This difference follows from the smaller time step in MLB, observed above in Figure 4 and Figure 9b. The MLB_λ method (Figure 13 in red) has smaller model error than for LB_k regardless of the choice for λ since MLB_λ benefits from the elimination of the chattering behaviour, without a reduction of time step for smaller entries.

An alternative approximation is $\lambda_{N_d} \approx N_d t_\ell^i \parallel r_\ell^i \parallel_\infty$, following from the observation that for $i \in \mathcal{S}$, one can also characterize the magnitude of $[\Delta_\ell G_\ell]^i$ in terms of $\Delta_\ell^i \parallel A_\ell^T \parallel \sigma$. For the test examples we consider here, $\sigma \approx \parallel b_\ell \parallel_2^{-1} \parallel r_\ell^i \parallel_2$ for $r_\ell^i = A_\ell x_\ell^i - b_\ell$ with $i \in \mathcal{S}$. Then for $\parallel x \parallel_2 = O(1)$, this alternative gives a reasonable approximation for Equation (5.1).

In general, the approximation Equation (5.1) is based on $\mathcal{S}$ which relies on prior knowledge of the noise level, usually not available in applications. In our examples, we have used the normalized residual to approximate the noise, which yields a good approximation to Equation (5.1) as described above for the alternative approximation for λ_{N_d}. Often in practice the threshold parameter is chosen using some naive estimations that do not involve the noise level or the number of data passes. These preliminary results using Equation (5.1) suggest that in the inconsistent setting, there can be value in approximating the noise level and identifying the desired number of data passes in order to accelerate convergence without overfitting the noise. Further details and improvements of the estimation of the parameter λ will be explored in future work. For example, the approximation in Equation (5.1) suggests an adaptive approximation to λ that can be used throughout the iterations.

6. Implications for large scale problems. So far, we discussed chatter in small over-determined but inconsistent toy problems. While this idealized setting has been widely studied, real life problems such as the motivating problem SPLS_σ are not strictly sparse but rather compressible. In that case, we hope to recover as many small entries in the tail of the solution as possible, given a fixed but limited number



(a) Iteration 21 using $\mathrm{LB_k}$ (b) Iteration 22 using $\mathrm{LB_k}$ (c) Iteration 23 using $\mathrm{LB_k}$

(d) Iteration 21, with MLB (e) Iteration 22, with MLB (f) Iteration 23, with MLB

Fig. 14: The resulting migration image for three consecutive iterations. Panels a)-c) repeat the results from Figure 2 using the $\mathrm{LB_k}$ method, where chattering is observed. Panels d)-f) shows the results using the MLB with the weighted increment of Equation (3.1). Note that the chattering behaviour of SPLS_σ is eliminated using MLB.

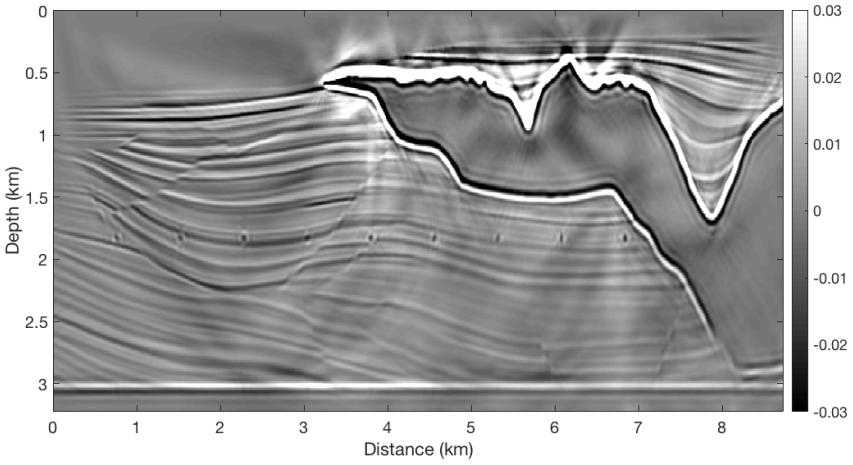
of iterations based on subsampling. To give a perspective with regard to the size of the motivating problem, we solved SPLS_σ on a 971×359 domain m with $n_s = 101$ different source experiments (or "shots"). Since we solve the PDE problem in a fine domain with many more points than in the background model m_0 and the number of "shots", thus the effective matrix of the PDE has many millions of rows.

Now we compare the results from Section 2, where $\mathrm{LB_k}$ was used for the motivating example, with results obtained using MLB for the same problem. Figure 14a, Figure 14b and Figure 14c show pronounced chattering behaviour when $\mathrm{LB_k}$ is used, and it is not possible to resolve the smaller entries. In Figure 14d, Figure 14e and Figure 14f the chattering behaviour has been eliminated when we use the MLB method. In the application of these methods, we used a subsampling that touches 10% of the "shots" in each iteration.

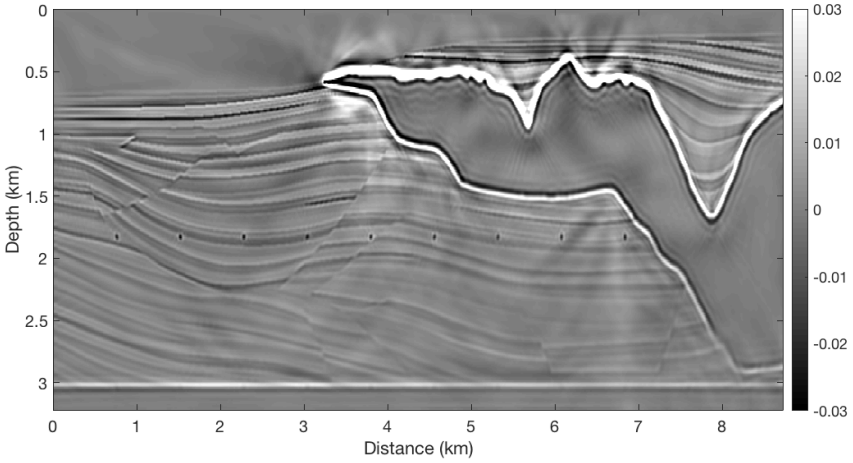
In Figure 15 we compare the resulting migration image at iteration 81 (equivalent to 8 data passes) for the $\mathrm{LB_k}$ method and the MLB. The difference is indeed quite visible: qualitatively the image using the $\mathrm{LB_k}$ method appears washed out, without the details obtained using MLB. The lack of resolution for $\mathrm{LB_k}$ is a manifestation of the stagnated model error, induced by the chattering behaviour. Since MLB eliminates the chattering behaviour, more of the details are resolved, resulting in a reduced model error.

Similar results to the MLB method can be obtain with the MLB_λ modification, also yielding eliminated chattering behaviour, with more details resolved as well. For this example the MLB_λ was marginally (1/3-1/2 of one data pass) faster in resolving the details on the model.

7. Conclusions and discussion. Sparse recovery has undoubtedly resulted in major breakthroughs in the field of compressive sensing. Yet straightforward adaptation of this technique towards the inversion of large over-determined, inconsistent



(a) Iteration 81 with the original linearized Bregman algorithm LB_{k}



(b) Iteration 81 with the modified linearized Bregman algorithm MLB with the weighted increment of Equation (3.1)

Fig. 15: The resulting migration image, using LB_{k} (upper) vs. using MLB (lower).

systems is challenged by stalling in the convergence of the algorithm, thus rendering this technique impractical for large-scale problems. By identifying this stalling behavior as a form of chatter, induced by the interplay between the stochastic dynamics of the iterations due to the inconsistency and the thresholding nonlinearity, we introduce simple counter measures via weighted increments and a entry-specific modification designed to control the chatter and accelerate convergence of the iterations. As a result, we arrive at a formulation capable of inverting extremely large inconsistent systems encountered in multi-experiment wave-equation based imaging problems. For each iteration, this type of imaging problem only gives us access to a few row blocks, which correspond to different source experiments. Despite this re-

striction, in combining the fast convergence of the linearized Bregman algorithm with the smart weighted increments that combat chatter, we have demonstrated that our method solves the problem in a few data passes (epochs).

Even though our inversion problem is essentially linear, its stochastic dynamics is intricate because there are multiple sources contributing to the fluctuations in the solution - the inconsistency, the randomized sampling, reminiscent of the widely employed stochastic gradient descent method, and the thresholding operation related to the sparsity objective. A critical step in our analysis is isolating the source of the undesirable chatter in order to remove it without unnecessarily limiting the solution space over which the algorithm searches. In that sense, our problem is similar to problems in machine learning where training is conducted with stochastic gradient descent on neural networks that consist of linear affine mappings intertwined with threshold-like operations. For this reason, we expect the ideas underlying the chatter counter measures proposed in this paper to extend to the training of (deep) neural nets.

REFERENCES

[1] J.-F. Cai, S. Osher, and Z. Shen, *Convergence of the linearized Bregman iteration for L_1-norm minimization*, Mathematics of Computation, 78 (2009), pp. 2127–2136.

[2] E. Candes and J. Romberg, *Sparsity and incoherence in compressive sampling*, Inverse problems, 23 (2007), p. 969.

[3] E. J. Candès, J. Romberg, and T. Tao, *Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information*, IEEE Transactions on information theory, 52 (2006), pp. 489–509.

[4] A. Cotter, O. Shamir, N. Srebro, and K. Sridharan, *Better mini-batch algorithms via accelerated gradient methods*, in Advances in Neural Information Processing Systems 24, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, eds., Curran Associates, Inc., 2011, pp. 1647–1655, http://papers.nips.cc/paper/4432-better-mini-batch-algorithms-via-accelerated-gradient-methods.pdf.

[5] E. Daskalakis, F. J. Herrmann, and R. Kuske, *Developments in the direction of solving extremely large problems in geophysics*, SEG Technical Program Expanded Abstracts, (2017), pp. 4375–4378.

[6] M. di Bernardo, A. R. Champneys, C. J. Budd, and P. Kowalczyk, *Piecewise-smooth Dynamical Systems: Theory and Applications*, Springer, 2008, https://doi.org/10.1007/978-1-84628-708-4.

[7] D. L. Donoho, *For most large underdetermined systems of linear equations the minimal l1-norm solution is also the sparsest solution*, Communications on Pure and Applied Mathematics, 59 (2006), pp. 797–829, https://doi.org/10.1002/cpa.20132, http://dx.doi.org/10.1002/cpa.20132.

[8] D. L. Donoho, A. Maleki, and A. Montanari, *Message-passing algorithms for compressed sensing*, Proceedings of the National Academy of Sciences, 106 (2009), pp. 18914–18919, https://doi.org/10.1073/pnas.0909892106, http://www.pnas.org/content/106/45/18914, https://arxiv.org/abs/http://www.pnas.org/content/106/45/18914.full.pdf.

[9] J. Duchi, E. Hazan, and Y. Singer, *Adaptive subgradient methods for online learning and stochastic optimization*, J. Mach. Learn. Res., 12 (2011), pp. 2121–2159, http://dl.acm.org/citation.cfm?id=1953048.2021068.

[10] G. Hennenfent, E. van den Berg, M. P. Friedlander, and F. J. Herrmann, *New insights into one-norm solvers from the Pareto curve*, Geophysics, 73 (2008), pp. A23–A26, https://doi.org/10.1190/1.2944169, https://www.slim.eos.ubc.ca/Publications/Public/Journals/Geophysics/2008/hennenfent08GEOnii/hennenfent08GEOnii.pdf.

[11] F. J. Herrmann, *Approximate message passing meets exploration seismology*, in 2012 IEEE Statistical Signal Processing Workshop (SSP), Aug 2012, pp. 25–28, https://doi.org/10.1109/SSP.2012.6319676.

[12] D. A. Lorenz, F. Schöpfer, and S. Wenger, *The linearized Bregman method via split feasibility problems: Analysis and generalizations*, SIAM Journal on Imaging Sciences, 7 (2014), pp. 1237–1262, https://doi.org/10.1137/130936269, http://dx.doi.org/10.1137/130936269, https://arxiv.org/abs/http://dx.doi.org/10.1137/130936269.

[13] D. A. Lorenz, S. Wenger, F. Schöpfer, and M. Magnor, *A sparse Kaczmarz solver and a linearized Bregman method for online compressed sensing*, in 2014 IEEE International Conference on Image Processing (ICIP), Oct 2014, pp. 1347–1351, https://doi.org/10.1109/ICIP.2014.7025269.

[14] A. Montanari, *Graphical models concepts in compressed sensing*, Cambridge University Press, 2012, p. 394–438, https://doi.org/10.1017/CBO9780511794308.010.

[15] A. Nedic and D. P. Bertsekas, *Incremental subgradient methods for nondifferentiable optimization*, SIAM Journal on Optimization, 12 (2001), pp. 109–138.

[16] D. Needell, *Randomized Kaczmarz solver for noisy linear systems*, BIT Numerical Mathematics, 50 (2010), pp. 395–403, https://doi.org/10.1007/s10543-010-0265-5, http://dx.doi.org/10.1007/s10543-010-0265-5.

[17] D. Needell and J. A. Tropp, *Paved with good intentions: Analysis of a randomized block Kaczmarz method*, Linear Algebra and its Applications, 441 (2014), pp. 199 – 221, https://doi.org/http://dx.doi.org/10.1016/j.laa.2012.12.022, http://www.sciencedirect.com/science/article/pii/S0024379513000098.

[18] D. J. W. Simpson, *Bifurcations in Piecewise-Smooth Continuous Systems*, World Scientific, 2010, https://doi.org/10.1142/7612, https://www.worldscientific.com/doi/abs/10.1142/7612, https://arxiv.org/abs/https://www.worldscientific.com/doi/pdf/10.1142/7612.

[19] T. Strohmer and R. Vershynin, *A randomized Kaczmarz algorithm with exponential convergence*, Journal of Fourier Analysis and Applications, 15 (2008), p. 262, https://doi.org/10.1007/s00041-008-9030-4, http://dx.doi.org/10.1007/s00041-008-9030-4.

[20] V. Utkin, J. Guldner, and J. Shi, *Sliding Mode Control in Electro-mechanical Systems*, CRC Press, 2009.

[21] E. van den Berg and M. Friedlander, *Sparse optimization with least-squares constraints*, SIAM Journal on Optimization, 21 (2011), pp. 1201–1229, https://doi.org/10.1137/100785028, https://doi.org/10.1137/100785028, https://arxiv.org/abs/https://doi.org/10.1137/100785028.

[22] E. van den Berg and M. P. Friedlander, *Probing the Pareto frontier for basis pursuit solutions*, SIAM Journal on Scientific Computing, 31 (2008), pp. 890–912, https://doi.org/10.1137/080714488, http://link.aip.org/link/?SCE/31/890.

[23] L. Wang, Y. Yang, R. Min, and S. Chakradhar, *Accelerating deep neural network training with inconsistent stochastic gradient descent*, Neural Networks, 93 (2017), pp. 219 – 229, https://doi.org/https://doi.org/10.1016/j.neunet.2017.06.003, http://www.sciencedirect.com/science/article/pii/S0893608017301399.

[24] Y. Wang and W. Yin, *Sparse signal reconstruction via iterative support detection*, SIAM J. Imaging Science, 3 (2010), pp. 462–491, https://doi.org/10.1137/090772447.

[25] M. Yang, P. Witte, Z. Fang, and F. Herrmann, *Time-domain sparsity-promoting least-squares migration with source estimation*, in SEG Technical Program Expanded Abstracts 2016, Society of Exploration Geophysicists, 2016, pp. 4225–4229.

[26] W. Yin, S. Osher, D. Goldfarb, and J. Darbon, *Bregman iterative algorithms for l_1-minimization with applications to compressed sensing*, SIAM Journal on Imaging Sciences, 1 (2008), pp. 143–168, https://doi.org/10.1137/070703983, http://dx.doi.org/10.1137/070703983, https://arxiv.org/abs/http://dx.doi.org/10.1137/070703983.

[27] H. Zou and T. Hastie, *Regularization and variable selection via the elastic net*, Journal of the Royal Statistical Society, Series B, 67 (2005), pp. 301–320.