HIGH-ORDER METHODS FOR BASIS PURSUIT

TOM GOLDSTEIN AND SIMON SETZER

Abstract. We introduce a second-order method for solving L1 regularized minimization problems. In most situations, this method finds the exact solution to an L1 regularized minimization in a finite number of steps. This is done by exploiting the fact that, when recovering a sparse signal, the basis pursuit problem is equivalent to a quadratic minimization problem involving a low-rank matrix. The novel method takes advantage of this property by applying a conjugate gradient partan scheme to the underlying low-rank problem. This new algorithm has several advantages over other basis-pursuit schemes. In particular, it requires no time-step parameters as input, and the speed of the algorithm is almost completely insensitive to the condition number of the problem. This allows for the fast solution of basis pursuit problems involving convolution matrices.

1. Introduction

Basis pursuit refers to the reconstruction of sparse signals by solving an L1 regularized problem of the following form:

$$\min_{u\in R^N} F(u) = \mu|u| + \frac{1}{2}\|Au - f\|^2 \tag{1}$$

where $A \in R^{M\times N}$ is a linear operator, f is a vector of observed data, $|\cdot|$ denotes the L1 norm, and $\|\cdot\|$ denotes the L2 norm.

Problems of this form arise frequently in compressed sensing(CS), where we wish to reconstruct u from a small subset of its Fourier/DCT coefficients [4, 5, 6, 10, 23, 27]. Problems of this form also arise in signal processing [8], analog-to-digital conversion [28, 15] and statistical regression [26].

Another class of problems that can be represented in the form (1) are sparse deconvolution problems. Sparse deconvolution problems have the form

$$\min_{u} \mu|u| + \frac{\mu}{2}\|Ku - f\|^2 \tag{2}$$

where K is a convolution matrix. These problems arise, for example, in heat-source identification, seismology, and medical imaging applications [17, 19, 20, 21, 25, 12, 11].

Problems of the form (1) are traditionally solved using techniques of the gradient-descent type. This is because fast algorithms can be used to evaluate the Fourier transform of u. Methods of the gradient descent type have several drawbacks. First, these methods require a-priori knowledge of the spectrum of A in order to choose an appropriate timestep. Also, gradient descent methods are often very sensitive to the condition number of the sensing matrix A.

In this manuscript, we proposed a method which uses conjugate gradient acceleration to improve the convergence of standard first-order methods. The organization

of this paper is as follows: We first review some common techniques for basis-pursuit problems. We then introduce the conjugate gradient partan method, and explain how it can be used to accelerate conventional forward-backward splitting techniques. We next introduce the CGIST method, which uses conjugate gradient acceleration to quickly solve problems of the form (1) without any knowledge of the spectrum of A. Finally, we show numerical results demonstrating the efficiency of the CGIST method.

2. Gradient-Based Methods for Basis Pursuit

In this section we review commonly used gradient-based methods for finding sparse solutions to systems of equations. Note that algorithms of this type only require that we be able to evaluate the linear operator A and its adjoint. In case the operator A involves a Fourier transform, step 3 of the FBS algorithm can be evaluated quickly using the fast Fourier transform (FFT). This makes gradient-based methods advantageous for problems involving fast transforms (e.g. Fourier, wavelet, DCT, etc...). Another advantage of the these methods is that they have simple implementation.

FBS. One of the simplest and most commonly used gradient-descent-type methods is the Forward-Backward Splitting (FBS). This approach has also been called iterative shrinkage/thresholding (IST) in the L1 literature. Algorithms of this form were originally proposed for by Lions and Mercier [16] and Passty [22], and were later studied extensively by others [9, 7, 18, 29]. Rigorous results for L1-regularized problems were proposed by Hale, Yin, and Zheng in [13].

Like other forward-backward splitting techniques, FBS is a two stage algorithm that operates on some initial guess u^k. During the first stage, we obtain $\bar{u}^k$ using a gradient descent step on the differentiable term in (1).

$$\bar{u^k} = u^k - tA^T(Au^k - f).$$

During the second stage, we update the value of $\bar{u}^k$ by solving the "proximal" problem

$$u^{k+1} = arg\min \mu|u| + \frac{1}{2t}\|u - \bar{u}^k\|$$

The overall algorithm can be written as

Algorithm 1 Forward-Backward Splitting (FBS)

1: Initialize: $u^0 \in R^N$
2: **for** $k = 0, 1, \cdots$ **do**
3: $\quad \bar{u}^k = u^k - tA^T(Au^k - f)$
4: $\quad u^{k+1} = arg\min \mu|u| + \frac{1}{2t}\|u - \bar{u}^k\|$
5: **end for**

For the problems considered here, the minimization in step 4 of the above algorithm has a simple closed form solution. For $L1$ regularized problems, we take

$$u^{k+1} = arg\min \mu|u| + \frac{1}{2t}\|u - \bar{u}^k\| = \frac{u}{|u|}\max\{u - t\mu, 0\} = shrink(\bar{u}^k, t\mu). \tag{3}$$

To see the relationship between FBS and gradient descent, consider the case when $sign(u^{k+1}) = sign(u^k)$. In this case, we have

$$u^{k+1} = u^k - t(\mu sign(u^k) + A^T(Au^k - f)) = u^k - t\partial\left\{\mu|u| + \frac{1}{2}\|Au - f\|^2\right\}|(u^k).$$

In simple words, when $sign(u^{k+1}) = sign(u^k)$, the non-differentiability of the energy (1) does not come into play, and the FBS algorithm is equivalent to marching down the gradient of (1). The relationship between FBS and gradient descent is the basis of many techniques for accelerating the algorithm.

2.1. **TwIST.** Several techniques have been proposed for accelerating the forward backward splitting. The first of these we shall consider is the Two-Step-Iterative-Shrinkage/Thresholding (TwIST) method, proposed and analyzed by Bioucas-Dias and Figueiredo [2, 3]. The TwIST algorithm is build from more conventional two-step methods for solving linear systems. These algorithms proceed in two stages: 1) a gradient descent step is performed, and 2) Over-relaxation is applied to accelerate convergence. When this concept is applied to the algorithm (1), we get a method of the following form:

Algorithm 2 TwIST

1: Initialize: $u^0 \in R^N$
2: $u^1 = shrink(u^0 - tA^T(Au^0 - f), t\mu)$
3: **for** $k = 0, 1, \cdots$ **do**
4: $\quad \bar{u}^k = shrink(u^k - tA^T(Au^k - f), t\mu)$
5: $\quad u^{k+1} = (1-\alpha)u^{k-1} + (\alpha - \beta)u^k + \beta\bar{u}^k$
6: **end for**

This algorithm can be interpreted as performing one FBS step, and then over-relaxing by adding the difference between the last two iterates.

It still remains to choose the relaxation parameters α and β in the algorithm (2). In the case that A is invertible, this can be done in the following way: Choose ξ_l and ξ_u to be a lower and upper bounds for the Eigenvalues of A^TA. Furthermore, define $\bar{\xi}_u = \max(\xi_u, 1)$, and $\kappa = \xi_l/\bar{\xi}_u$. The authors of [2] then suggest to choose

$$\alpha = \left(\frac{1-\sqrt{\kappa}}{1+\sqrt{\kappa}}\right)^2 + 1, \qquad \beta = \frac{2\alpha}{\bar{\xi}_u + \xi_l}.$$

For the case of invertible A, it can be shown that these parameter choices guarantee convergence. When A is a non-invertible matrix, this rule breaks down and parameters may have to be chosen by hand.

The authors of [2] also suggest that the asymptotic convergence rate of the TwIST algorithm is substantially faster than FBS. Using FBS, the number of iterations required to reduce the error by an order of magnitude is approximately $-\log_{10}\frac{1-\kappa}{1+\kappa}$. On the other hand, TwIST takes approximately $-\log_{10}\frac{1-\sqrt{\kappa}}{1+\sqrt{\kappa}}$ steps to achieve the same error reduction, which is considerably faster FBS.

2.2. **FPC_AS.** All algorithms discussed so far are first order methods. A method which incorporated higher-order information is the FPC_AS algorithm [31]. The FPC_AS method is based on the standard forward-backward splitting technique. FPC_AS, however, does not rely on a fixed timestep. Rather, a Barzilai-Borwein

rule is used to adaptively choose the stepsize [1]. In addition, when the active set stabilizes, a conjugate gradient or BFGS based subspace optimization routine is called in order to speed up convergence. This subspace optimization step makes FPC_AS a very efficient choice for ill-conditioned problems, such as deconvolutions, for which first-order methods tend to stall. However, for well-conditioned problems (e.g. compressed sensing) the overhead of the subspace optimization step can make this method slow. For a detailed description of subspace optimization and adaptive timestepping for FBS, we refer the reader to [31].

Note that FPC_AS was originally designed to solve problems with small μ by using a continuation process - e.g. by solving a sequence of problems with decreasing values of μ^k, such that $\mu^k \to \mu$). For comparison purposes, the continuation portion of the algorithm was deactivated for this study.

3. Gradient PARTAN Algorithm

One of the most effective ways of minimizing quadratic energies is the method of conjugate gradients (CG) [14]. The CG method in its standard form, however, cannot be directly viewed as a means of accelerating gradient descent routines. The reason for this is that the standard CG algorithm requires that descent steps be taken in directions which are not gradients. This is fine when solving standard quadratic minimization problems because we are free to move in any direction. For basis pursuit problems, however, this does not work so well. It is difficult to descend in arbitrary directions because we encounter the non-differentiable points of the energy (1). Furthermore, in order to be able to take consistent descent steps without regularizing the energy (1) we need to be able to use the shrinkage formula (3), which only allows us to move along the gradient of the energy (1).

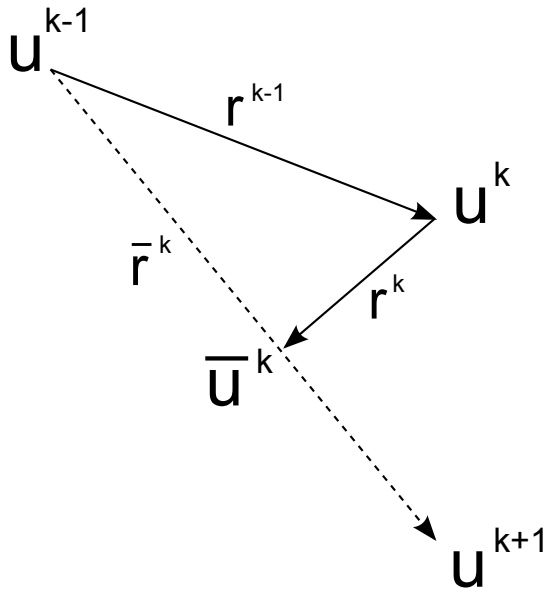
Fortunately, there exists a special conjugate gradient routine which only requires moves along gradient directions. This variant of CG, introduced by Shah, Buehler, and Kempthorne, is called gradient partan [24].

An iteration of the partan algorithm consists of two stages: Starting with an approximate solution u^k, we 1) Find the minimum point, $\bar{u}^k$, of the energy in the gradient direction. 2) Find the minimizer, u^{k+1}, of the energy in the direction of the vector $\bar{r} = u^{k-1} - \bar{u}^k$. This process is depicted diagrammatically in Figure 3.

We formalize this algorithm below. Note that the partan algorithm was originally designed to minimize arbitrary convex quadratic energies. The version of the method presented below has been reformulated to solve problems of the special form

$$\min_u \frac{1}{2}\|Au - f\|^2 + \langle u, s\rangle \tag{4}$$

FIGURE 1. A diagrammatic depiction of the gradient partan algorithm. Starting from point u^k, we first compute the gradient vector, r^k. We then find the minimizer of the energy, $\bar{u}^k$ along this direction. The next approximation, u^{k+1}, is then obtained by minimizing in the direction $\bar{r} = u^{k-1} - \bar{u}^k$.



Algorithm 3 Partan Algorithm for solving (4)

Initialize: $u^0 = 0$
$r^0 = A^T(Au^0 - f) + s$
$\alpha^0 = \frac{\|r^0\|^2}{\|Ar^0\|^2}$
$u^1 = u^0 - \alpha^0 r^0$
$r^1 = r^0 + \alpha^0 A^T A r^0$
for $k = 1, 2, 3, \cdots$ **do**
  $\alpha^k = \frac{\|r^k\|^2}{\|Ar^k\|^2}$
  $\bar{u}^k = u^k - \alpha^k r^k$
  $\bar{r}^k = r^k + \alpha^k A^T A r^k$
  $\beta^n = \frac{\langle \bar{r}^k, r^{k-1} \rangle}{\|r^{k-1}\|^2}$
  $u^{k+1} = \frac{\hat{u}^k - \beta^k u^{k-1}}{(1-\beta)}$
  $r^{k+1} = \frac{\hat{r}^k - \beta^k r^{k-1}}{(1-\beta)}$
end for

At the start of the algorithm, two iterates are initialized because the conjugate gradient acceleration step requires two successive iterates. Once the "for" loop has been entered, the algorithm proceeds to minimize the energy (4) along the direction of the gradient vector, r^k. This is done by first computing the optimal step-length,

α^k, and then taking the corresponding gradient descent step. This minimizer is denoted $\bar{u}^k$. The next step of the algorithm is to minimize the energy along the line between $\bar{u}^k$ and u^{k-1}. This is done by first computing the optimal step size parameter, β. The minimizer, u^{k+1}, in this new direction is then computed using a linear combination of $\bar{u}^k$ and u^{k-1}.

This partan algorithm has several valuable properties for solving quadratic minimization problems that the two-step methods do not. Because it is a conjugate gradient routine, partan converges to the exact minimizer in a number of steps which is less than or equal to the dimension of the problem. For problems with large dimension, a good approximation to the solution is often found quickly because partan (like all CG algorithms) has super-linear convergence properties that other gradient-descent methods do not. Finally, the partan algorithm does not require the user to choose any time step parameters that depend on the (possibly unknown) spectral properties of the matrix A.

3.1. **The Contribution of This Paper.** As noted above, forward-backward splitting methods for L1 regularized problems behave like gradient descent for quadratic problems. *The key observation we wish to exploit is that, when solving sparse problems, the dimensionality of the gradient descent problem is very small.* For example, if there are only 5 non-zero elements in an iterate of the FBS method, then the method behaves like gradient descent on a problem with dimension 5. In this case, the conjugate gradient routine described above could solve the minimization exactly in at most 5 steps, regardless of the condition number of A. Gradient descent, on the other hand, could take hundreds of steps if the matrix A is ill-conditioned.

In this manuscript we will develop a method which exploits this low-dimensionality property using the conjugate gradient partan method. This is done by adding an acceleration term to the FBS iteration. This new term will allow the algorithm to minimize the energy (1) over its active set in a finite number of steps.

4. High-Order Methods for Basis Pursuit

In this section, we will show how to interpret problems of the form (1) in such a way that algorithm (3) can be applied. We introduce two new algorithms. The first uses an adaptive rule to choose an optimal step length. The second uses conjugate gradient acceleration in addition to the adaptive step size rule. Because these methods work by applying conjugate gradient acceleration to iterative shrinkage/thresholding (i.e. FBS) methods, we call the new algorithms CGIST0 and CGIST, respectively.

4.1. **Reformulation as a Low-Rank Problem.** Suppose that for two successive iterates of the FBS method we have $sign(u^{k-1}) = sign(u^k)$. Let $D = (u^{k-1}! = 0)$ be the diagonal matrix with $D_{ii} = 1$ if $u^{k-1} \neq 0$, and $D_{ii} = 0$ otherwise. Then the process of obtaining u^k from u^{k-1} using FBS is equivalent to a gradient descent step on the differentiable quadratic problem

$$u^k = arg \min_{u \in R^N} \frac{1}{2}\|Au - f\|^2 + \langle u, s\rangle, \text{ such that } Du = 0, \tag{5}$$

where $s = \mu sign(u^{k-1})$.

We now wish to remove the constraint $Du = 0$. Note that, when this constraint is satisfied, we have $u = Du$. We can therefore replace the constrained problem (5)

with the problem

$$u^k = arg \min_{u \in R^N} \|ADu - f\|^2 + \langle u, s \rangle \tag{6}$$

The energy (6) has derivative

$$r = DA^T(ADu - f) + s \tag{7}$$

Note that, if $u_i^{k-1} = 0$, then the derivative of the energy (6) with respect to u_i is zero. It follows that a gradient descent step on the energy (6) will give us $u_i^k = u_i^{k-1} = 0$. Therefore, provided $sign(u^{k-1}) = sign(u^k)$, gradient descent on (6) is equivalent to FBS on (1).

Note the dimensionality reduction that has occurred by reformulating the basis-pursuit problem (1) in the form (6). The rank of the matrix AD is less than or equal to the number of non-zeros in the iterate u^{k-1}. Consequently, if u^{k-1} is sparse, then we would expect a conjugate gradient method to solve (6) exactly in a small number of steps.

We can write the derivative of (6) in a more convenient way. We have

$$r_i = \begin{cases} (DA^T(ADu - f) + s)_i, & \text{if } u_i \neq 0 \\ shrink(DA^T(ADu - f)_i, \mu), & \text{otherwise} \end{cases},$$

or equivalently

$$r = DA^T(ADu - f) + s + (I - D)shrink(A^T(Au - f), \mu). \tag{8}$$

Suppose that for some index i we have $u_i^k = u_i^{k+1} = 0$. Then we must have $shrink(A^T(Au - f), \mu)_i = 0$. It follows that, when the active set does not change, the two expressions for the derivative (8) and (7) are equivalent.

The derivative definition (8) is advantageous because it is nonzero for every non-optimal u. To see this, suppose that we have $r = 0$. Then we have

$$\begin{cases} (DA^T(ADu - f) + s)_i = 0, & \text{if } u_i \neq 0 \\ (DA^T(ADu - f))_i \in [0, 1], & \text{if } u_i = 0 \end{cases}$$

which is the optimality condition for (1). Later on, will use the derivative (8) of the quadratic problem in order to compute the optimal step size for the FBS iteration. Using the definition (8) will help prevent this technique from breaking down.

4.2. **Optimal Adaptive Time-stepping.** Before we can apply the conjugate gradient partan method to the basis pursuit problem, we must establish a method of performing gradient descent. Note that, when performing a gradient descent step, there is no guarantee that $sign(u^k) = sign(u^{k-1})$. For this reason, we must perform gradient descent steps using the FBS algorithm (which incorporates a shrinkage operator). This guarantees that, in the case when the active set changes, our iterates still decrease the energy (1).

We therefore wish to compute the gradient step using the FBS formula

$$g^k = A^T(Au - f) \tag{9}$$

$$\bar{u}^k = shrink(u^k - \alpha g^k, \alpha\mu) \tag{10}$$

where α denotes the stepsize. We will choose the optimal step size such that the quadratic energy (6) is minimized in the gradient direction. To compute this step size, we must use the gradient of the quadratic energy as defined by equation (8).

Using the definition g^k given above, this derivative can be written in the condensed form

$$r^k = D^k g^k + \mu sign(u^k) + (I - D^k) shrink(g^k, \mu). \tag{11}$$

The optimal step size which minimizes the quadratic energy is then given by

$$\alpha^k = \frac{\|r^k\|^2}{\|Ar^k\|^2} \tag{12}$$

When we incorporate this step size rule into the FBS algorithm, we get the following method:

Algorithm 4 CGIST0: FBS with Optimal Adaptive Time Steps

1: Initialize: u^0
2: **for** $k = 1, 2, 3, \cdots$ **do**
3: $\quad g^k = A^T(Au^k - f)$
4: $\quad D^k = (u^k \neq 0)$
5: $\quad r^k = D^k g^k + \mu sign(u^k) + (1 - D^k) shrink(g^k, \mu)$
6: $\quad \alpha^k = \frac{\|r^k\|^2}{\|Ar^k\|^2}$
7: $\quad u^{k+1} = shrink(u^k - \alpha^k g^k, \alpha^k \mu)$
8: **end for**

Note that algorithm (4) does not require any time-step parameters as inputs. This can be very advantageous when A is an unstructured matrix and we have little information about the spectrum of $A^T A$ to choose a time-step. Also note that the adaptive algorithm (4) requires 3 matrix multiplications per iteration, as opposed to 2 per iteration for conventional FBS. We will see later that, despite this extra cost, the optimal time-stepping method (4) substantially outperforms conventional methods for many problems.

4.3. **Conjugate Gradient Acceleration.** We now wish to apply conjugate gradient acceleration to the FBS method by exploiting its equivalence to the quadratic problem (6). We will do this using the partan gradient descent method (3). The complete algorithm with conjugate gradient acceleration is listed in Algorithm 5. This algorithm has a more sophisticated sequence of steps than the previous algorithms we have discussed, and we will address the function of each of these steps below.

Algorithm 5 CGIST: High Order Algorithm for Basis Pursuit

Initialize: u^0
$D^0 = (u^0 \neq 0)$
$e^0 = Au - f$
$g^0 = A^T e^0$
$r^0 = D^0 g^0 + \mu sign(u^0)) + (1 - D^0) shrink(g^0, \mu)$
$\alpha^0 = \frac{\|r^0\|^2}{\|Ar^0\|^2}$
$u^1 = shrink(u^0 - \alpha^0 g^0, \alpha^0 \mu)$
$D^1 = (u^1 \neq 0)$
$e^1 = Au^1 - f$
$g^1 = A^T e^1$
for $k = 1, 2, 3, \cdots$ **do**
  $r^k = D^k g^k + \mu sign(u^k)) + (1 - D^k) shrink(g^k, \mu)$
  $\alpha^k = \frac{\|r^k\|^2}{\|Ar^k\|^2}$
  $\bar{u}^k = shrink(u^k - \alpha^k g^k, \alpha\mu)$
  if $sign(\bar{u}^k) == sign(u^k) == sign(u^{k-1})$ **then**
    $\bar{e}^k = e^k - \alpha A r^k$
    $\bar{g}^k = A^T \bar{e}^k$
    $\bar{r}^k = r^k + D^k(\bar{g}^k + g^k)$
    $\bar{\beta}^k = \frac{\langle \bar{r}^k, r^{k-1} \rangle}{\|r^{k-1}\|^2}$
    $\beta^k = \min(\bar{\beta}^k, D^k \bar{u}^k / u^{k-1})$
    $u^{k+1} = \frac{\bar{u}^k - \beta^k u^{k-1}}{(1-\beta^k)}$
    $e^{k+1} = \frac{\bar{e}^k - \beta^k e^{k-1}}{(1-\beta^k)}$
    $D^{k+1} = (u^{k+1} \neq 0)$
    $g^{k+1} = \frac{\bar{g}^k - \beta^k g^{k-1}}{(1-\beta^k)}$
    $r^{k+1} = \frac{\bar{r}^k - \beta^k r^{k-1}}{(1-\beta^k)}$
  else
    $u^{k+1} = \bar{u}^k$
    $D^{k+1} = (u^k \neq 0)$
    $e^{k+1} = Au^{k+1} - f$
    $g^{k+1} = A^T e^{k+1}$
    $r^k = D^{k+1} g^{k+1} + \mu sign(u^{k+1})) + (1 - D^{k+1}) shrink(g^{k+1}, \mu)$
  end if
end for

Because the acceleration step in the partan algorithm requires knowledge of the two most recent iterates, we must initialize two iterates before entering the loop. This initialization is done by starting with an arbitrary u^0, and then applying a single iteration of FBS with optimal time step. The algorithm then enters the "for" loop. The first action taken in the loop is to compute $\bar{u}^k$ using an FBS step with optimal parameter.

Note that the problems (1) and (6) are only equivalent if $sign(u^k) = sign(u^{k+1})$. Consequently, we only wish to apply the conjugate gradient acceleration when this condition is satisfied. Once $\bar{u}^k$ has been computed, the first "if" statement tests whether $sign(u^k) = sign(\bar{u}^k)$. If this condition fails, then we accept $u^{k+1} = \bar{u}^k$ as

our next iterate, and continue to the next iteration of the "for" loop. In this case, it is also necessary to re-compute the gradient g^{k+1} to be used in the next iteration.

If the compatibility condition is satisfied, then the active set of our problem has not changed, and the algorithm proceeds to compute the conjugate gradient acceleration step. The algorithm computes the gradient at the current approximation, and the constant $\bar{\beta}^k$ just as is done in the partan algorithm (3). However, our acceleration formula is only correct as long as the active set of our problem remains constant. To ensure that the acceleration step does not alter the active set, we must compute the maximal value of β for which $sign(u^{k+1}) = sign(\bar{u}^{k+1})$. It is easily seen that the active set does not change as long as $\beta \leq \bar{u}^k/u^{k-1}(u^{k+1}! = 0)$. We therefore define $\min(\bar{\beta}^k, \bar{u}^k/u^{k-1}(u^{k+1}! = 0))$ to ensure that the active set does not change. Finally, the values of u^{k+1}, g^{k+1}, and r^{k+1} are computed for the next iterate.

5. Convergence Results

In this section we address two issues related to the convergence of the above methods. First, we discuss conditions that must be met for the CGIST/CGIST0 methods to be guaranteed to converge. Next, we discuss the finite convergence property of the CGIST method.

The methods described above use the conjugate gradient step size to perform FBS. In the case that the active set does not change, this step size can be shown to be optimal. However, when the active set does change, this rule is purely heuristic and there is no guarantee of optimality. For this reason, some care must be taken in order to guarantee convergence of the above algorithms for all possible inputs. In this section, we discuss a simple modification of the above methods that can be used to guarantee convergence. These modifications are mostly of theoretical interest, as we have found both algorithms 4 and 5 to be extremely stable.

Convergence of algorithms 4 and 5 can be guaranteed by incorporating a non-monotonic line search of Wen, Yin, Goldfarb and Zheng [31, 30]. Given the current iterate u^k, we first generate $\bar{u}^k$ using a step of FBS with the conjugate gradient step size α^k. We then define the search direction $d^k = \bar{u}^k - u^k$, and choose a $0 < \gamma < 1$ such that $u^{k+1} = u^k + \alpha d^k$ does not significantly increase the energy (1). This is done using Algorithm (6).

Algorithm 6 Backtracking Line Search

Initialize: $u^k, \bar{u}^k, \nu \in [0,1), C^0 \geq F(u^0), Q^0 = 1, c \in (0,1)$
$Q^k = \nu Q^{k-1} + 1$
$C^k = \frac{Q^k - 1}{Q^k} C^{k-1} + \frac{1}{Q^k} F(u^{k-1})$
$d^k = \bar{u}^k - u^k$
$\Delta^k = {g^k}^T d^k + \mu(|\bar{u}^k| - |u^k|)$
$\gamma = 1$
while $F(u^k + \gamma d^k) > C^k + c\gamma\Delta^k$ **do**
    $\gamma \leftarrow \gamma/2$
end while
Set $u^{k+1} = u^k + \gamma d^k$

The application of this line search to FBS-type methods is considered in [30], where a convergence result is given. Note that the line search does not force each iteration to decrease the objective function. Rather, the energy of each iteration must be sufficiently less than some upper threshold, C^k, which evolves by a recursive formula. The parameter ν controls how severely each iterate can violate the monotonicity condition. Note that the line search can be made monotone by letting $\nu = 0$. Non-monotone line search is often preferable to monotone line search because it is less likely that the line search will become active.

The next issue that must be addressed is that the step size rule (12) can break down if $Ar^k = 0$. In this case, we can simply perform an FBS step with step size equal to unity, and then apply the backtracking line search to guarantee a decrease in energy. We emphasize that these precautions are only necessary in extreme circumstances, as the empirical behavior of algorithms 4 and 5 is very stable. In fact, the line search was never activated in any of the numerical examples considered in this manuscript.

The next issue we address is the finite convergence of the CGIST algorithm. This finite convergence property arises because of the rank reduction property discussed in section 4.1. When the active set of the problem remains constant, the energy behaves like a low rank quadratic energy. In this case, the conjugate gradient method minimizes the energy over its active set in a finite number of steps. This concept is formalized below:

Theorem. *Suppose that the CGIST algorithm (5) converges to a minimizer of (1). Then the algorithm obtains an exact minimizer in a finite number of steps.*

Proof. Let $u^* = arg\min_u F(u)$. We begin by proving the following claim: There exists an $\epsilon > 0$ such that, for any u satisfying $F(u) < F(u^*) + \epsilon$, we have $sign(u) = sign(u^*)$. Stated more simply, we wish to prove that any u with sufficiently small energy must have the same active set as the true minimizer u^*. Since $sign(u)$ can only take on finitely many values, this will be true as long as there does not exist another minimizer $\bar{u}^*$ with $sign(u^*) \neq sign(\bar{u}^*)$.

We therefore assume for contradiction that we have distinct minimizers satisfying $sign(u^*) \neq sign(\bar{u}^*)$. Consider the vector $\hat{u} = \frac{1}{2}(u^* + \bar{u}^*)$. Since the differentiable (i.e. L2) part of the energy (1) is weakly convex, we have

$$\|A\hat{u} - f\|^2 \leq \frac{1}{2}(\|A\bar{u}^* - f\|^2 + \|Au^* - f\|^2). \tag{13}$$

Also, since $sign(u^*) \neq sign(\bar{u}^*)$ we have the strict inequality

$$|\hat{u}| < \frac{1}{2}(|u^*| + |\bar{u}^*|). \tag{14}$$

Adding (13) to (14) yields

$$F(\hat{u}) < \frac{1}{2}(F(u^*) + F(\bar{u}^*)) = F(u^*)$$

which is a contradiction since u^* is a minimizer. This, together with the remarks above, shows the validity of the claim.

It follows that, once the objective function gets sufficiently small, the active set of the iterates of CGIST remains constant. Once the active set of the iterates stabilizes, CGIST reduces to the standard conjugate gradient partan method for quadratic objectives, which is known to converge in a finite number of steps [14, 24]. □

Note that in the above proof, we have neglected the effect of roundoff errors on the conjugate gradient method. For large-rank problems, these errors can accumulate, causing the conjugate gradient method to behave more like an iterative method than an exact solver. However, because of the rank reduction property described in section 4.1, the effective rank of sparse reconstruction problems tends to be very small. For this reason, we have found that the finite convergence is observed in practice.

6. Numerical Results

To demonstrate the performance of this algorithm, we compare it to several other L1 minimization algorithms. For this purpose, we will use three categories of test problems. The first type of problem involves the discrete cosine transform (DCT-II). Problems of this type are compressed sensing problems, for which the sensing matrix $A \in R^{M\times N}$ is formed by selecting M rows of the discrete cosine transform matrix. The goal of the problem is to recover a K-sparse signal of length N. The sparsity pattern, as well as the row sampling, are generated uniformly at random. For the trials reported below, we choose $N = 1000$, $N = 200$, and $K = 20$. Recovery was performed with regularization parameter $\mu = 0.04$. The measurements were contaminated with Gaussian noise of standard deviation $\sigma = 0.01$ before recovery.

The second problem category is once again a compressed sensing problem. However, this time we use a random Gaussian matrix – i.e. a matrix whose elements are drawn from a Gaussian distribution with zero mean and unit variance. The problem is to reconstruct a K-sparse signal from an MxN Gaussian matrix, with $N = 1000$, $M = 100$, and $K = 10$. Recovery was performed with regularization parameter $\mu = 10$. Measurements were contaminated with noise of standard deviation $\sigma = 0.01$.

The final problem we consider is an ill conditioned deconvolution problems. The goal of these problems is to recover a 20-sparse vector of length N from convolution data. The sensing matrix for this application is of the form RC, where C denotes the N-point cosine transform, and R is a diagonal matrix with $R_{ii} = i^{-0.7}$. The fidelity parameter for this problem is $\mu = 0.002$. Because deconvolution results are highly sensitive to noise, the measurement were contaminated with random noise with standard deviation $\sigma = 10^{-5}$.

The CGIST algorithm is compared to FBS, TwIST, and FPC_AS. The FBS algorithm was performed using a constant time step equal to $1.9/\rho(A^TA)$. Parameters for the TwIST algorithm were chosen to be $\alpha = 1.9$, and $\beta = 3.9$, which are similar to those suggested in [2, 3]. In the results displayed below, "CGIST" refers to the partan accelerated algorithm 5, while "CGIST0" refers to the non-accelerated version, algorithm 4. The results displayed below were created by generating 100 random instances of each problem type, and averaging results over all trials.

Note that the FBS and TwIST algorithms require knowledge of the spectrum of A in order to choose a stable stepsize. For this reason, FBS and TwIST require a setup phase in which the eigenvalues of A are pre-computed. This setup phase is frequently much more costly than solving the underlying L1 minimization. However, this setup phase can in some situations be precomputed and stored for later use, or amortized over the cost of many similar L1 minimizations. For this reason, we do not report this setup time in the results below.

Convergence curves for all three problems are displayed in Figure 2. The horizontal axis in these curves displays the total number of matrix multiplications needed to attain each level of convergence. By "Matrix multiplications," we mean the number of times the algorithm has evaluated the linear sensing operator, A, or its transpose. We have found that for all algorithms considered here, virtually all of the computation time is consumed by evaluating the sensing operator. For this reason, the number of matrix multiplications is an accurate way of comparing the performance of each method. Furthermore, matrix multiplications provide a measure of performance that is independent of the computer, operating system, and programming language used.

For all three problems considered here, the CGIST and CGIST0 algorithms out performed the other methods considered. The performance advantage of CGIST was particularly apparent for problems involving the DCT. For the other two problems, however, there is another algorithm with similar convergence properties. For Gaussian matrices, the performance gap between CGIST and TwIST did not become prominent until high levels of precision were attained. Also, for the deconvolution problem, FPC_AS is almost as efficient as CGIST. It was found that the CGIST and CGIST0 methods exhibit similar performance until relatively high levels of precision are attained (e.g. $< 1\%$ relative error). However, once a certain level of precision is attained, CGIST accelerates substantially and outperforms CGIST0.

The compressed sensing problem is investigated further in Figure 3. Here, the number of iterations needed to attain a relative error below 10^{-6} is plotted for various sparsity levels. It can be seen that for this problem, the high-order CGIST/CGIST0 and FPC_AS methods consistently outperform the lower order methods. As the number of non-zero elements increases, the CGIST computing time seems to scale slightly better than the other high-order schemes. For large numbers of non-zeros, the FBS method was by far the slowest of the schemes.

For a quantitative performance comparison, Table 1 displays the number of matrix multiplications needed to reduce the relative error below 10^{-6} for a variety of problems.

FIGURE 2. Convergence curves for the FBS, TwIST, CGIST0, CGIST, and FPC_AS methods. The vertical axis measures the log error, defined as $\log(\|u^k - u^*\|_2)$. Convergence curves are shown for the DCT (top), Gaussian (center), and deconvolution (bottom) problems.

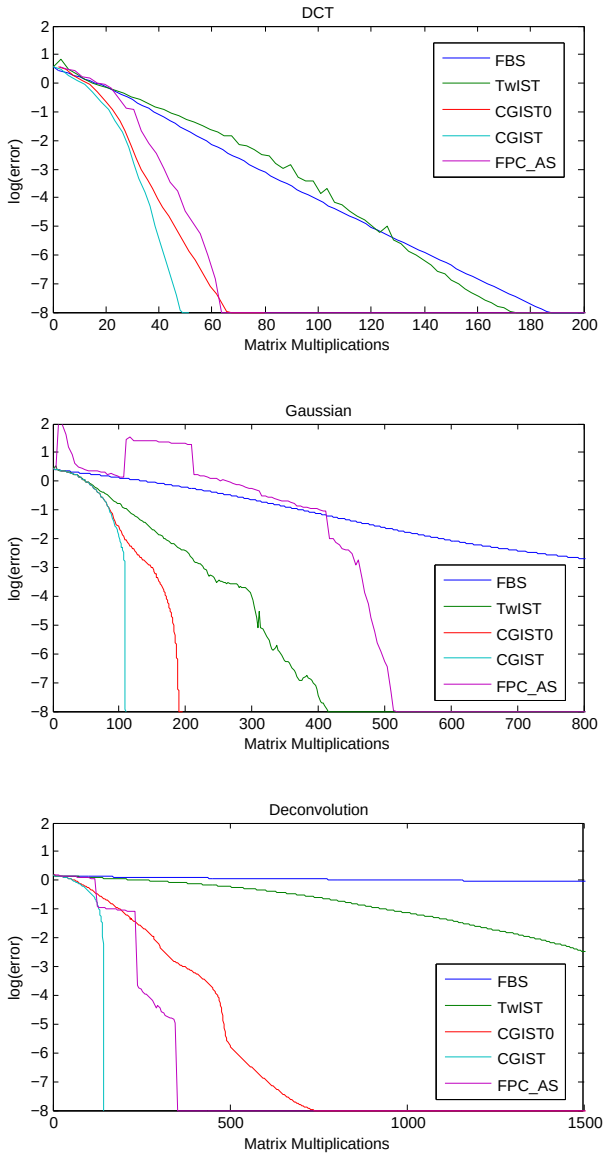


FIGURE 3. Number of iterations required to reach 10^{-6} relative error as a function of sparsity, for the compressed sensing problem. For the compressed sensing problem, it was found that CGIST method remained the most efficient as the number of non-zeros increased.

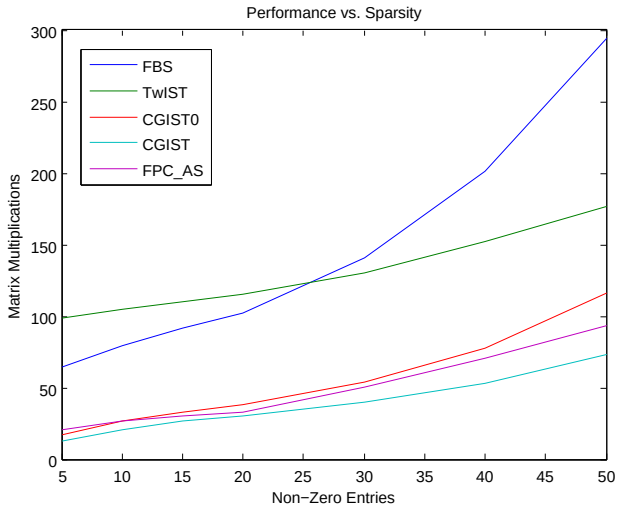


TABLE 1. Matrix multiplications needed to attain 10^{-6} relative error

Algorithm	DCT	Gaussian	Deconvolution
FBS	126.9	824.8	1980.2
TwIST	131.2	209.8	1168.0
FPC_AS	47.9	316.3	182.5
CGIST0	48.1	121.8	226.8
CGIST	38.0	100.4	144.2

7. Conclusion

We introduce a new set of methods for solving L1-regularized minimization problems. Unlike conventional first order methods, the newly proposed methods use high order information to achieve extremely fast convergence rates. Because conjugate-gradient acceleration is used, the methods also exploit the low-rank properties of sparse vectors, allowing L1 problems to be solved exactly in finitely many steps. The proposed schemes are compared to several other methods, demonstrating the efficiency of the high-order approach.

References

[1] Jonathan Barzilai and Jonathan M. Borwein. Two-point step size gradient methods. *IMA J Numer Anal*, 8(1):141–148, January 1988.

[2] J. Bioucas-Dias and M. Figueiredo. A new TwIST: Two-step iterative shrinkage/thresholding algorithms for image restoration. *IEEE Transactions on Image Processing*, 16(12):2992–3004, Dec. 2007.

[3] J.M. Bioucas-Dias and M.A.T. Figueiredo. Two-step algorithms for linear inverse problems with non-quadratic regularization. *IEEE International Conference on Image processing*, 1:105–108, Sept 2007.

[4] E. J. Candes, J. Romberg, and T.Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inform. Theory*, 52:489 – 509, 2006.

[5] E. J. Candes and T. Tao. Near-optimal signal recovery from random projections: universal encoding strategies? *IEEE Transactions on Information Theory*, 52(12):5406–5425, 2006.

[6] Emmanuel J. Candes and Justin Romberg. Quantitative robust uncertainty principles and optimally sparse decompositions. *Found. Comput. Math.*, 6(2):227–254, 2006.

[7] George H.-G. Chen and R. T. Rockafellar. Convergence rates in forward–backward splitting. *SIAM J. on Optimization*, 7(2):421–444, 1997.

[8] Scott Shaobing Chen, David L. Donoho, and Michael A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20:33–61, 1998.

[9] Patrick L. Combettes and Valérie R. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling and Simulation*, 4(4):1168–1200, 2005.

[10] David L. Donoho. Compressed sensing. *IEEE Trans. Inform. Theory*, 52:1289–1306, 2006.

[11] Willard S. Ellis, Susan J. Eisenberg, David M. Auslander, Michael W. Dae, Avideh Zakhor, and Michael D. Lesh. Deconvolution: A novel signal processing approach for determining activation time from fractionated electrograms and detecting infarcted tissue. *Circulation*, 94:2633–2640, 1996.

[12] N.P. Galatsanos, A.K. Katsaggelos, R.T. Chin, and A.D. Hillery. Least squares restoration of multichannel images. *IEEE Transactions on Signal Processing*, 39:2222–2236, 1991.

[13] Elaine Hale, Wotao Yin, and Yin Zhang. A fixed-point continuation method for l1-regularized minimization with applications to compressed sensing. *CAAM Technical Report*, TR07, 2007.

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

[15] Sami Kirolos, Jason Laska, Michael Wakin, Marco Duarte, Dror Baron, Tamer Ragheb, Yehia Massoud, and Richard Baraniuk. Analog-to-information conversion via random demodulation. In *IEEE Dallas/CAS Workshop on Design, Applications, Integration and Software*, 2006.

[16] P. L. Lions and B. Mercier. Splitting algorithms for the sum of two nonlinear operators. *SIAM Journal on Numerical Analysis*, 16(6):964–979, 1979.

[17] J.M. Mendel and C.S. Currus. *Maximum-likelihood deconvolution: a journey into model-based signal processing*. Springer-Verlag, 1990.

[18] Muhammad Aslam Noor. Splitting algorithms for general pseudomonotone mixed variational inequalities. *J. of Global Optimization*, 18(1):75–89, 2000.

[19] M.S. O'Brien, A.N. Sinclair, and S.M. Kramer. Recovery of a sparse spike time series by l1 norm deconvolution. *IEEE Transactions on Signal Processing*, 42:3353–3365, 1994.

[20] T. Olofsson and E. Wennerstrom. Sparse deconvolution of b-scan images. *IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control*, 54:1634–1641, 2007.

[21] Tomas Olofsson. Semi-sparse deconvolution robust to uncertainties in the impulse responses. *Ultrasonics*, 37:423–432, 1999.

[22] B. G Passty. Ergodic convergence to a zero of the sum of monotone operators in hilbert space. *Journal of Mathematical Analysis and Applications*, 72:386–390, 1979.

[23] Mark Rudelson and Roman Vershynin. Geometric approach to error correcting codes and reconstruction of signals. *Int. Math. Res. Not*, 64:4019–4041, 2005.

[24] B. V. Shah, R. J. Buehler, and O Kempthorne. Some algorithms for minimizing a function of several variables. *J. Society for Industrial and Applied Mathematics*, 12:74–92, 1964.

[25] Oloffson T. and Stepinksi T. Maximum a posteriori deconvolution of sparse ultrasonic signals using genetic optimization. *Ultrasonics*, 37:423–432, 1999.

[26] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.

[27] J.A. Tropp. Just relax: convex programming methods for identifying sparse signals in noise. *Information Theory, IEEE Transactions on*, 52:1030–1051, 2006.

[28] J.A. Tropp, M.B. Wakin, M.F. Duarte, D. Baron, and R.G. Baraniuk. Random filters for compressive sampling and reconstruction. *IEEE International Conference on Acoustics, Speech and Signal Processing*, 3, 2006.

[29] Paul Tseng. A modified forward-backward splitting method for maximal monotone mappings. *SIAM J. Control Optim*, 38:431–446, 1998.

[30] Zaiwen Wen, Wotao Yin, and Donald Goldfarb. On the convergence of an active set method for l1 minimization. Technical report, Rice University, 2010.

[31] Zaiwen Wen, Wotao Yin, Donald Goldfarb, and Yin Zheng. A fast algorithm for sparse reconstruction based on shrinkage, subspace optimization, and continuation. Technical report, Rice University, 2009.