A MULTILEVEL DECOMPOSITION METHOD FOR SPARSE SIGNAL RECOVERY IN A FOURIER BASIS

TOM GOLDSTEIN

ABSTRACT. We introduce a multi-level decomposition scheme for solving basis pursuit problems in a Fourier basis. The iterates generated by this scheme are equivalent to the coordinate-descent (CD) method for basis pursuit. However, unlike standard CD methods, the new algorithm computes each iterate using only $O(n \log n)$ operations. Using four different test problems, the CD algorithm is compared to two other common algorithms for basis pursuit. For the problems considered here, runtimes for the CD algorithm are approximately 5-10 times faster than conventional methods.

1. INTRODUCTION

In this manuscript, we consider a fast multi-level decomposition method for reconstructing sparse signals from samples taken in the Fourier domain. In particular, we wish to solve problems of the form

$$\min_{u \in R^N} \sum_{i=0}^{N-1} \phi(u_i) + \sum_{i=0}^{N-1} \frac{\mu}{2} (R_i \mathcal{F}_N u_i - s_i)^2 \tag{1}$$

where $\{R_i\}$ is some sequence on real numbers, s_i is the observed data, ϕ is some regularizing function, and $\mathcal{F}_N$ represents the discrete Fourier transform operator defined on vectors of size N. This energy can be written more compactly using vector norm notation as

$$\min_{u \in R^N} \Phi(u) + \frac{\mu}{2} \|R\mathcal{F}_N u - s\|^2 \tag{2}$$

where R denotes the diagonal matrix having the sequence $\{R_i\}$ as its diagonal entries, and $\|\cdot\|$ denotes the discrete 2-norm.

Problems of this form arise frequently in compressed sensing(CS), where we wish to reconstruct u from a small subset of its Fourier coefficients [6, 7, 8, 13, 33, 40]. If the kth Fourier coefficient is known, we take s_k to be the coefficient, and $R_k = 1$. If the kth mode is unknown, we take $s_k = R_k = 0$. Problems of this form also arise in signal processing [10], analog-to-digital conversion [41, 21] and statistical regression [37].

Another class of problems that can be represented in the form (2) are sparse deconvolution problems. Sparse deconvolution problem have the form

$$\min_u |u| + \frac{\mu}{2} \|Ku - s\|^2 \tag{3}$$

where K is a convolution matrix. These problems arise, for example, in heat-source identification, seismology, and medical imaging applications [25, 27, 28, 29, 34, 19, 16]. Problems of the form (3) can be written in the form (4) by noting that

$K = \mathcal{F}_N^* R \mathcal{F}_N$ for some diagonal matrix R. Because of the unitary nature of the Fourier transform, we have

$$\|Ku - s\|^2 = \|\mathcal{F}_N^* R \mathcal{F}_N u - s\|^2 = \|R\mathcal{F}_N u - \mathcal{F}_N s\|^2$$

where the rightmost form of this energy is the same as that appearing in (4).

Many different choices of ϕ have been suggested in order to enforce sparsity of the recovered signal. One of the most common choices is the L1 regularizer $\phi(u_i) = |u_i|$. This regularizer has the advantage of being (weakly) convex, making it possible to efficiently compute a global minimizer.

Another technique for enforcing sparsity is to force the recovered signal to take on only positive values. This leads to the "non-negative least squares" technique. Non-negative least squares corresponds to problems of the form (2) where we choose

$$\phi^+(u) = \begin{cases} 0 & \text{if } u \geq 0 \\ \infty & \text{if } u < 0 \end{cases}$$

The uniqueness and compressed sensing properties of non-negative least squares are studied in [5, 14].

Problems of the form (2) 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. In this manuscript, we propose to solve problems of the form (2) using a multi-level decomposition scheme which is equivalent to coordinate descent. The organization of this paper is as follows: We first review some common techniques for basis-pursuit problems including the coordinate descent method. We then show how the coordinate descent method can be efficiently applied to problems involving the Fourier transform. We also describe how the coordinate descent method can incorporate inequality constraints (such as non-negativity) which would be difficult to incorporate into gradient based methods. Finally, we show numerical results demonstrating the efficiency of coordinate descent based methods for these problems.

2. Background

In this section we review commonly used methods for finding sparse solutions to systems of equations. These algorithms apply to problems of the form

$$\min_{u \in R^N} \Phi(u) + \frac{\mu}{2}\|Au - s\|^2 \tag{4}$$

where $A : R^N \to R^M$ is an arbitrary linear operator.

FPC. A very common approach to solving problems of the form (4) is to use a gradient-descent-based method such as Fixed-Point Continuation (FPC) [20]. This technique is based on the concept of forward-backward splitting. Algorithms of this form were originally proposed for differentiable problems by Lions and Mercier [23] and Passty [31], and later studied extensively by others [9, 26, 45]. Rigorous results for L1-regularized problems were first proposed by Hale, Yin, and Zheng in [20]. Techniques for accelerating this simple iteration scheme have also been proposed and analyzed in [3, 1], although these variants will not be considered here.

Like other forward-backward splitting techniques, FPC is a two stage algorithm that operates on some initial guess u^k. During the first stage, we obtain u^k using

a gradient descent step on the differentiable term in (4).

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

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

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

The overall algorithm can be written

Algorithm 1 Fixed-Point Continuation (FPC)

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

$$u_{k+1} = arg\min |u| + \frac{\mu}{2t}\|u - \bar{u^k}\| = \frac{u}{|u|}\max\{u - \frac{t}{\mu}, 0\}.$$

In the case of non-negative least squares, we take

$$u_{k+1} = arg\min \Phi^+ + \frac{\mu}{2t}\|u - \bar{u^k}\| = \max\{u, 0\}.$$

Note that the FPC algorithm (1) only requires 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 FPC algorithm can be evaluated quickly using the fast Fourier transform (FFT). This makes FPC advantageous for problems involving the Fourier transform (and other fast transforms). Another advantage of the FPC method is its extremely simple implementation.

Orthogonal Matching Pursuit. One way to enforce sparsity is to explicitly penalize non-zero entries using a penalty of the form

$$\Phi^0(u) = \|u\|_0,$$

where the "$L0$ norm" counts the number of non-zero entries of u. Because such a regularizer is non-convex, the resulting minimization problem is not in general computationally tractable. However, it is possible to attempt to solve the problem using heuristic methods, such as orthogonal matching pursuit (OMP) [24, 35, 36, 42].

OMP is a greedy algorithm in which elements are added to the support of u one at a time until a suitable approximation to the sparse signal is reached. The algorithm is initialized with $support(u^0) = \emptyset$. Then, a single element is added to the support of u^1 in order to minimize the resulting residual error, $\|s - Au^1\|_2^2$. On each successive iteration, the support of the signal is expanded by one element, so that $|support(u^k)| = k$. This is formalized in algorithm (2). Note that we have used a_i to denote the ith column of A.

Algorithm 2 Orthogonal Matching Pursuit (OMP)

Intialize: $u^0 = 0$
$S^k = support(u^k) = \emptyset$
for $k = 0, 1, \cdots$ **do**
  $r^k = s - Au^k$
  for $i = 1, 2, \cdots, N$ **do**
    $\epsilon^i = \min_{z \in R} \|r^k - a_i z\|$
  end for
  Choose i^* such that $\forall i$, $\epsilon_{i^*} \leq \epsilon_i$
  $S^{k+1} = S^k \cup \{i^*\}$
  $u^{k+1} = arg\min_{support(u) \in S^{k+1}} \|s - Au\|^2$
end for

The outer loop in algorithm (2) is usually terminated when $\|s - Au^k\|$ falls below a preset tolerance. Although the $L0$ penalized problem is non-convex, it has been shown that, in the context of compressed sensing problems with sufficiently sparse signals, this algorithm will find a global minimum with reasonably high probability [24, 42]. For such problems it has also been shown that the solutions to the $L0$ penalized problem and the $L1$ penalized problem will coincide with reasonably high probability. For this reason, the OMP algorithm can be considered as a substitute for (or at least an approximation to) the $L1$ regularized problem.

Coordinate Descent. Fixed Point Continuation is a gradient descent based minimization technique. For general basis pursuit problems (e.g. problems not involving the Fourier transform or other fast transforms), faster methods are available. For unconstrained problems, coordinate descent (CD) is frequently the most efficient approach. These techniques work by minimizing (2) with respect to each individual element u_i in sequence.

Algorithm 3 Coordinate Descent (CD)

Intialize: $u^0 \in R^N$
for $k = 0, 1, \cdots$ **do**
  for $i = 1, 2, \cdots, N$ **do**
    $u_i^k \leftarrow arg\min_{u_i} \Phi(u) + \frac{\mu}{2}\|Au - s\|$
  end for
end for

Methods of this type were proposed very early by Fu [18], and were later popularized by Daubechies et al. [12]. Variants of this algorithm have been studied extensively for various applications including the elastic net [48, 46], non-negative least squares [4], grouped regression [47], and many other applications [22, 44, 43]. Variants of this algorithm have also been proposed for TV regularized problems (also called the "fused lasso") by Tibshirani and others [38, 39, 2, 30]. A detailed review of many of these techniques can be found in [17].

Not only does CD have a faster convergence rate than FPC, but for dense A the cost of a CD iteration is lower then the cost of an FPC iteration. These two factors make CD a superior solver when speed is the primary consideration.

Unfortunately, CD is generally not available for basis pursuit problems involving the Fourier transform. The reason for this is that CD iterations require $O(n^2)$ computations. FPC, on the other hand, require only matrix multiplications, which can be accomplished in $O(N \log N)$ operations using the Fast Fourier Transform.

In this manuscript, we introduce an algorithm that efficiently performs CD iterations on problems of the form (2) without using any explicit Fourier transforms. After introducing the algorithm, we will present numerical results demonstrating that this approach outperforms conventional methods by a considerable margin for a wide variety of applications.

3. The Cooley-Tukey FFT

One of the most efficient schemes for computing the FFT is the Cooley-Tukey factorization [11, 15, 32]. Although this type of procedure can be performed on vectors of arbitrary composite size, we will assume for simplicity that the signal length is a power of 2. In this case, we get a radix 2 decimation-in-time algorithm in which the Fourier transform of a length N signal is represented as a linear combination of smaller Fourier transforms of size $n = N/2$.

If we let $\omega_N = e^{\frac{-2\pi i}{N}}$ denote the Nth root of unity and $\mathcal{F}_N$ denote the FFT of size N, then this decomposition can be written in summation form as

$$\begin{aligned} \mathcal{F}_N u(k) &= \sum_{m=0}^{N-1} u_m \omega^{mk} = \sum_{m=0}^{n-1} u_{2m} \omega_N^{(2m)k} + \sum_{m=0}^{n-1} u_{2m+1} \omega_N^{(2m+1)(k)} \\ &= \mathcal{F}_n u_e(k) + e^{-\frac{2\pi i}{N}k} \mathcal{F}_n u_o(k) \end{aligned}$$

where u_e represents the even-indexed components of the u, and u_0 represents the odd-indexed components of u. To be more precise

$$u_e(m) = u(2m) \text{ and } u_o(m) = u(2m+1).$$

This structure of this algorithm is more intuitive when it is written as a matrix factorization. We write

$$\mathcal{F}_n u = \begin{pmatrix} I_n & D_N \\ I_n & -D_N \end{pmatrix} \begin{pmatrix} \mathcal{F}_n u_e \\ \mathcal{F}_n u_o \end{pmatrix}. \tag{5}$$

In the above factorization, D_N represents the diagonal matrix of "twiddle factors,"

$$D_{N,ii} = e^{-\frac{2\pi i}{N}k}$$

Using the Cooley-Tukey concept, we have written the Fourier matrix as a product of a block diagonal "butterfly matrix," and a set of smaller Fourier matrices. Because of the block-diagonal structure of the butterfly matrix, multiplication by this matrix can be performed in $O(N)$ operations. The Fourier transforms of the smaller matrices are performed by recursively applying the Cooley-Tukey formula. A simple induction argument shows that the total cost of computing the N-point FFT using this scheme is $O(N \log N)$

4. CD Minimization for Problems Involving the Fourier Transform

In this section, we discuss how to perform fast CD minimization for the basis pursuit problem (2). We will assume for simplicity that the signal length is a power of 2. Our method will work by subdividing this problem into two smaller problems,

solving these small problems, and then recombining the solutions. In this sense, the method is a divide-and-conquer algorithm much like the FFT itself.

We begin by re-writing the energy E_n in terms of the even and odd-indexed components of u. The element-wise decoupled L1 regularizing term can be easily decomposed this way. When we perform this decomposition, the energy (2) becomes

$$E_N(u) = E_N(u_e, u_o) = |u_e| + |u_o| + \frac{\mu}{2} F(u_e, u_o) \tag{6}$$

where

$$F(u_e, u_o) = \|R\mathcal{F}_N u - s\|^2. \tag{7}$$

To proceed with CD minimization, we must find a mechanism to decouple the even- and odd-indexed components of u in the energy (7). This is provided by the following theorem.

Theorem. *The energy (7) can be written in the following two equivalent forms*

$$\text{Form 1: } F(u_e, u_o) = \|R_0 F u_e - s_e\|^2 + C_e \tag{8}$$

$$\text{Form 2: } F(u_e, u_o) = \|R_0 F u_o - s_o\|^2 + C_o \tag{9}$$

where

$$\begin{aligned} R_0 &= (R_1^* R_1 + R_2^* R_2)^{1/2} \\ s_e &= R_1^* R_d^{-1} s_1 + R_2^* R_d^{-1} s_2 + (R_2^* R_2 - R_1^* R_1) R_d^{-1} D_N \mathcal{F}_{N/2} u_o \\ s_o &= D_N^* R_1^* R_d^{-1} s_1 - D_N^* R_2^* R_d^{-1} s_2 + (R_2^* R_2 - R_1^* R_1) R_d^{-1} D_N^* \mathcal{F}_{N/2} u_e \end{aligned}$$

and C_e does not depend on u_e, while C_o does not depend on u_o. We also have that R_d is the diagonal matrix

$$(R_d)_{ii} = \begin{cases} (R_0)_{ii} & \text{if } (R_0)_{ii} \neq 0 \\ 1 & \text{otherwise} \end{cases}$$

Proof. To show that (8) is equivalent to (7), we will show that both (8) and (7) have the same subgradient with respect to u_e. Using the Cooley-Tukey factorization (5), we can expand the energy (7) in terms of u_e and u_o. We get

$$F(u_e, u_o) = \left\| \begin{pmatrix} R_1 & 0 \\ 0 & R_2 \end{pmatrix} \begin{pmatrix} I_n & D_N \\ I_n & -D_N \end{pmatrix} \begin{pmatrix} \mathcal{F}_n u_e \\ \mathcal{F}_n u_o \end{pmatrix} - \begin{pmatrix} s_1 \\ s_2 \end{pmatrix} \right\|^2. \tag{10}$$

Equation (10) can be written

$$F(u_e, u_o) = \|R_1 \mathcal{F}_{N/2} u_e + R_1 D_N \mathcal{F}_{N/2} u_o - s_1\|^2 + \|R_2 \mathcal{F}_{N/2} u_e - R_2 D_N \mathcal{F}_{N/2} u_o - s_2\|^2$$

We then differentiate with respect to u_e, and rearrange the result:

$$\partial_e F = \mathcal{F}_n^* R_1^* (R_1 \mathcal{F}_n u_e + R_1 D_N \mathcal{F}_n u_o - s_1) \tag{11}$$

$$+\mathcal{F}_n^* R_2^* (R_2 \mathcal{F}_n u_e - R_2 D_N \mathcal{F}_n u_o - s_1) \tag{12}$$

$$= \mathcal{F}_n^* (R_1^* R_1 + R_2^* R_2) \mathcal{F}_n u_e + (\mathcal{F}_n^* R_1^* R_1 D_N \mathcal{F}_n - \mathcal{F}_n^* R_2^* R_2 D_N \mathcal{F}_n) u_o \tag{13}$$

$$-\mathcal{F}_n^* R_1^* s_1 - \mathcal{F}_n^* R_2^* s_2 \tag{14}$$

We next differentiate (8), which gives us

$$\partial_e F = \mathcal{F}_n^* R_0^* R_0 \mathcal{F}_n u_e - \mathcal{F}_n R_0^* R_d^{-1} (R_1^* R_1 - R_2^* R_2) D_N \mathcal{F}_n u_o \tag{15}$$

$$-\mathcal{F}_n^* R_0^* R_d^{-1} R_1^* s_1 - \mathcal{F}_n^* R_0^* R_d^{-1} R_2^* s_2 \tag{16}$$

Now, note that $(R_0)_{ii} = 0 \Leftrightarrow (R_1)_{ii} = (R_2)_{ii} = 0$. From this, and the definition of R_0 we get the following identities

$$R_0^* R_0 = R_1^* R_1 + R_2^* R_2 \qquad R_0^* R_d^{-1} R_1 = R_1 \qquad R_0^* R_d^{-1} R_2 = R_2.$$

When these identities are applied to (15-16), we see that it is equivalent to (13-14).

The proof of (9) follows from a similar argument. □

We now describe how the two forms of (10) can be used to perform element-wise minimization. Suppose we first want to perform element-wise minimization on the even-indexed elements of u. During this minimization step, the elements of u_o are fixed. The minimization problem we wish to solve can thus be written

$$\arg\min_{u_e \in R^n} E(u) = \arg\min_{u_e \in R^n} |u_e| + |u_o| + \frac{\mu}{2} F(u_e, u_o) = \arg\min_{u_e \in R^n} |u_e| + \frac{\mu}{2} F(u_e, u_o)$$

Because u_o is held constant, we choose to expand the energy F in the form (8) above. This gives us the equivalent problem

$$\arg\min_{u_e \in R^n} |u_e| + \frac{\mu}{2} \|R_0 F u_e - s_e\|^2.$$

Performing element-wise minimization of the energy (2) on the even index elements of u is thus equivalent to performing element-wise minimization on (4). Note that the constant factor C_e has been omitted here because it depends only on the constant vector u_o, and thus has no effect of the solution of (4).

Similarly, to perform element-wise minimization of the odd-indexed components of u, we need only perform a minimization sweep on the following problem of size $n = N/2$:

$$\arg\min_{u_u \in R^n} |u_u| + \frac{\mu}{2} \|R_0 F u_o - s_o\|^2.$$

The minimization of the small (size $n = N/2$) problem is performed recursively using the same decomposition that we used for problems of size N. On each stage of the recursion, the problem size gets reduced by a factor of 2. The recursion terminates when the problem size has been reduced to 1, and we must solve the resulting problem analytically. The length 1 problem has the form

$$\arg\min_{u \in R} |u| + \frac{\mu}{2} \|R \mathcal{F}_1 u - s\|^2 \tag{17}$$

If we note that $\mathcal{F}_1 = I_1$, we can see that this problem is easily solvable for many choices of ϕ. In particular, we have

$$\arg\min_{u \in R} |u| + \frac{\mu}{2} \|Ru - s\|^2 = \frac{x}{|x|} \max\{|x| - \frac{1}{\mu |R|^2}, 0\}. \tag{18}$$

5. Implementation of the Element-Wise method

Following the arguments above, element-wise minimization on (2) can thus be achieved by the following recursive algorithm:

Algorithm 4 : cdfft(v, R, s, μ, N) (slow version)

if $N = 1$ **then**
  return $shrink(s, 1/\mu|R|^2)$
end if
for $m = 0$ to $N/2 - 1$ **do**
  $u_e(m) = u(2m)$, and $u_o(m) = u(2m + 1)$
end for
Let $s_e = R_1^* R_d^{-1} s_1 + R_2^* R_d^{-1} s_2 + (R_2^* R_2 - R_1^* R_1) R_d^{-1} D_N \mathcal{F}_{N/2} u_o$
Recursive call: $u_e = \mathbf{cdfft}(u_e, R_0, s_e, mu, N/2)$
$s_o = D_N^* R_1^* R_d^{-1} s_1 - D_N^* R_2^* R_d^{-1} s_2 + (R_2^* R_2 - R_1^* R_1) R_d^{-1} D_N^* \mathcal{F}_{N/2} u_e$
Recursive call: $u_o = \mathbf{cdfft}(u_o, R_0, s_o, mu, N/2)$
for $m = 0$ to $N/2 - 1$ **do**
  $u(2m) = u_e(m)$, and $u(2m + 1) = u_o(m)$
end for
return u

This algorithm is slow, however, because it requires the computation of FFT's at each level (an FFT is involved in the definition of s_e and s_o). A close analysis of (4) shows that these FFT's can be eliminated by operating on the Fourier transform of u rather than u itself.

In order to do this, we make the substitution $v = \mathcal{F}_n u$. Our goal is to re-write algorithm (4) in terms of the variable v so that all computations can be done in the Fourier domain. For this purpose, we decompose v into its upper and lower halves

$$v = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$$

Now, using the Cooley-Tukey factorization (5), it is trivial to derive the following identities:

$$\begin{aligned} v_e &= \mathcal{F}_n u_e = \frac{1}{2}(v_1 + v_2) \\ v_o &= \mathcal{F}_n u_o = \frac{1}{2} D_N^* (v_1 - v_2) \\ v &= \begin{pmatrix} v_e + D_N v_o \\ v_e - D_N v_o \end{pmatrix} \end{aligned}$$

Using these identities, we can re-write the above algorithm so that we only operate in the Fourier domain:

Algorithm 5 : cdfft(v, R, s, μ, N)

1: **if** $N = 1$ **then**
2: **return** $shrink(s, 1/\mu|R|^2)$
3: **end if**
4: $v_e = \frac{1}{2}(v_1 + v_2)$
5: $v_o = \frac{1}{2}D_N^*(v_1 - v_2)$
6: $s_e = R_1^* R_d^{-1} s_1 + R_2^* R_d^{-1} s_2 + (R_2^* R_2 - R_1^* R_1) R_d^{-1} D_N v_o$
7: Recursive call: $v_e = \mathbf{cdfft}(v_e, R_0, s_e, mu, N/2)$
8: $s_o = D_N^* R_1^* R_d^{-1} s_1 - D_N^* R_2^* R_d^{-1} s_2 + (R_2^* R_2 - R_1^* R_1) R_d^{-1} D_N^* v_e$
9: Recursive call: $v_o = \mathbf{cdfft}(v_o, R_0, s_o, mu, N/2)$
10: $v = \begin{pmatrix} v_e + D_N v_o \\ v_e - D_N v_o \end{pmatrix}$
11: **return** v

The above algorithm requires no explicit FFT's, and runs in $O(n \log N)$ operations. Almost all of the computation takes place in steps 7 and 9, where we build the vectors s_e and s_o. Note that, while these formulas look bulky, they actually represent only a small amount of computation. Each of s_e and s_o are formed using a simple linear combination of 3 vectors, and the coefficients of this linear combination are precomputed only once.

6. Numerical Experiments

To demonstrate the performance of the CD algorithm, and compare it to FPC and OMP, we use four simple test problems: two problems from compressed sensing, and two deconvolution problems. These problems are summarized in table 1.

The first two test problems are conventional compressed sensing problems. We wish to recover a sparse signal from a subset of its Fourier modes. For each trial, a sparse signal of length 256 is generated by randomly choosing 5 non-zero elements of unit intensity. Problem CS1 is to recovery the signal using only 32 of its 256 Fourier coefficients, chosen at random. Problem CS2 recovers the signal using 128 randomly chosen Fourier coefficents. For both problems, we enforce sparsity using an L1-regularized problem of the form (4), with $\mu = 20$. The FPC and CD iterations were stopped when the condition $\|u^{k+1} - u^k\| < 10^{-8}$ was met. The OMP algorithm was terminated with the criteria $\|Au^k - s\| < 0.1$.

The second test problem we consider is a sparse deconvolution problem. In this case, the vector u^* represents the summation of 5 delta-function "sources" with unknown locations. The locations of these sources is chosen at random on each trial. The observed data, s, represents the summation of these delta functions after blurring with a Gaussian kernel. The goal of this problem is to "reverse the heat equation," and find the location of the unknown sources. This deconvolution problem is regularized with an L1 penalty to ensure sparse results. Problem D1 is a deconvolution problem with a Gaussian kernel of variance 10 pixels. Problem D2 is blurred with a kernel of variance 0.5. Both the FPC and CD algorithms were terminated when the condition $\|u^{k+1} - u^k\| < 10^{-4}$ was met.

Results averaged over 100 trials are reported in table 2. In addition to reporting the number of iterations and total runtime (in milliseconds) for each algorithm, the

TABLE 1. Four Numerical Test Problems

Problem Name	Type	Measurement
CS1	Compressed Sensing	32/256 Fourier Coefficients
CS2	Compressed Sensing	128/256 Fourier Coefficients
D1	Deconvolution	Gaussian Blur, $\sigma^2 = 10$
D2	Deconvolution	Gaussian Blur, $\sigma^2 = 0.5$

table also identifies the number of wrong atoms (i.e. non-zero elements) detected. The OMP method is not well suited for deconvolution problems, and it was found to generate excessively large numbers of wrong atoms for these problems. For this reason, we do not report OMP results for problems D1 and D2.

TABLE 2. Results

Problem	*Algorithm*	*Iterations*	*Runtime (ms)*	*Wrong Atoms*
	OMP	5.46	6.7	0.67
CS1	FPC	167	19.0	0.53
	CD	17.6	**1.2**	0.53
	OMP	4.96	5.5	0
CS2	FPC	20.5	2.4	0
	CD	8.53	**0.60**	0
	OMP	*	*	*
D1	FPC	3217	251.7	0.85
	CD	942	**55.3**	0.85
	OMP	*	*	*
D2	FPC	26.4	2.2	0
	CD	2.15	**0.18**	0

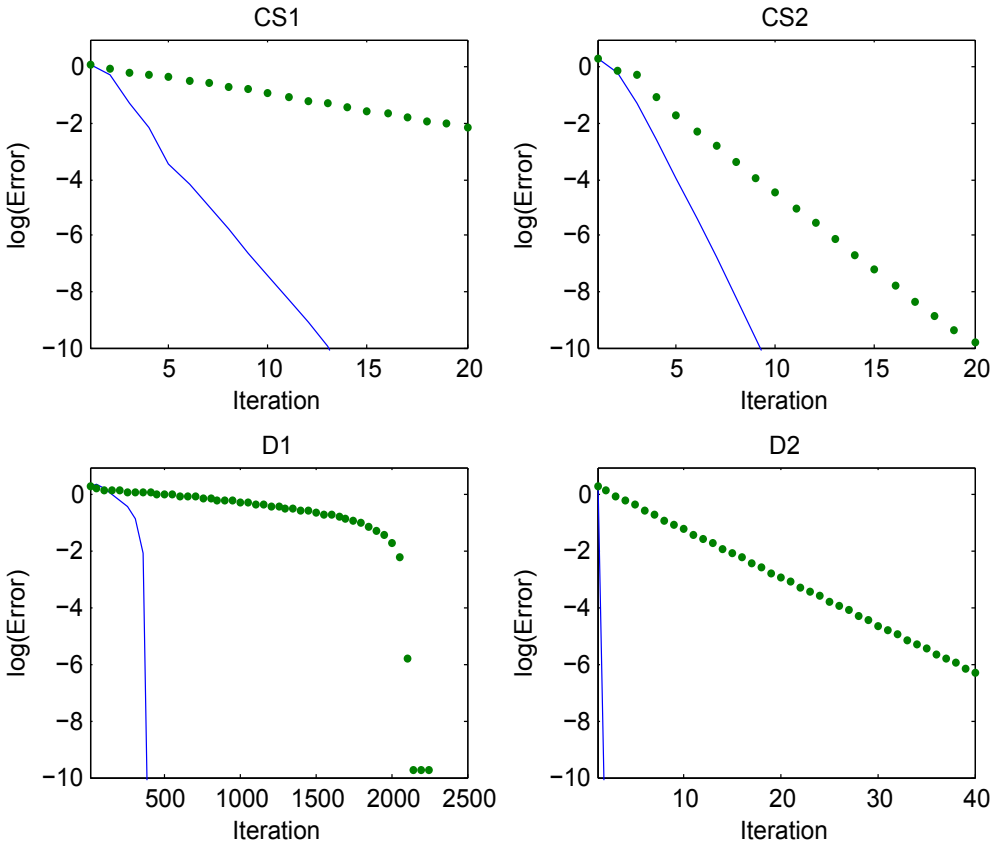
From the results in table 2, it is clear that one advantage of the coordinate descent (CD) method is speed. For the test problems considered here, it was observed that the CD method was approximately one order of magnitude faster than the OMP and FPC methods. Furthermore, for the ill-conditioned problem CS1, the L1-regularized methods tend to identify fewer wrong atoms then the L0 regularized OMP method. Another significant advantage of the CD method is that, unlike the FPC algorithm, it does not require the user to choose a time step and it has no stability restriction.

One notably disadvantage of the CD scheme is that its implementation is relatively complex when compared to FPC. The FPC method is extremely easy to implement in Matlab, and its implementation can take advantage of extremely well-optimized implementations of the fast Fourier transform, such as FFTW. In order to compete with such efficient codes, the CD algorithm described above must be implemented in a low-level language such as C/C++.

7. CONCLUSION

We introduce a multi-level decomposition scheme for solving basis pursuit problems in a Fourier basis. The iterates generated by this scheme are equivalent to

FIGURE 1. Convergence Curves for the four problems considered. Curves show log(*error*) vs. iteration number for the FPC and CD algorithms. Error is defined in the L2 sense, i.e. $error = \|u^k - u^*\|$. The CD algorithm is depicted by the solid blue line. The dotted green line represents the FPC algorithm.



the coordinate-descent (CD) method for basis pursuit. Using four different test problems, the CD algorithm is compared to two other common algorithms for basis pursuit. For the problems considered here, runtimes for the CD algorithm are approximately 5-10 times faster than conventional methods.

REFERENCES

[1] Stephen Becker, Jerome Bobin, and Emmanuel Candes. Nesta: A fast and accurate first-order method for sparse recovery. *Technical Report of Stanford University*, Apr 2009.
[2] Dimitri P. Bertsekas. *Nonlinear Programming*. Athena Scientific, September 1999.
[3] Bioucas. A new twist: Two-step iterative shrinkage/thresholding algorithms for image restoration. *Image Processing, IEEE Transactions on*, 16(12):2992–3004, Dec. 2007.
[4] Leo Breiman. Better subset regression using the nonnegative garrote. *Technometrics*, 37(4):373–384, 1995.

[5] A.M. Bruckstein, M. Elad, and M. Zibulevsky. On the uniqueness of nonnegative sparse solutions to underdetermined systems of equations. *Information Theory, IEEE Transactions on*, 54:4813–4820, 2008.
[6] 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.
[7] 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.
[8] Emmanuel J. Candes and Justin Romberg. Quantitative robust uncertainty principles and optimally sparse decompositions. *Found. Comput. Math.*, 6(2):227–254, 2006.
[9] George H.-G. Chen and R. T. Rockafellar. Convergence rates in forward–backward splitting. *SIAM J. on Optimization*, 7(2):421–444, 1997.
[10] Scott Shaobing Chen, David L. Donoho, and Michael A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20:33–61, 1998.
[11] J. W. Cooley and J. W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of computation*, 19(90):297–301, 1965.
[12] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics*, 57(11):1413–1457, 2004.
[13] David L. Donoho. Compressed sensing. *IEEE Trans. Inform. Theory*, 52:1289–1306, 2006.
[14] D.L. Donoho and J Tanner. Sparse nonnegative solutions of underdetermined linear equations by linear programming. *Proceedings of the National Academy of Sciences*, 102(27):9446–9451, 2005.
[15] P. Duhamel and M. Vetterli. Fast fourier transforms: a tutorial review and a state of the art. *Signal Process.*, 19(4):259–299, April 1990.
[16] 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.
[17] Jerome Friedman, Trevor Hastie, Holger Höfling, and Robert Tibshirani. Pathwise coordinate optimization. *Technical report of Dept. of Statistics, Stanford University*, Dec 2007.
[18] W. Fu. Penalized regressions: The bridge vs the lasso. *JCGS*, 7:397–416, 1998.
[19] 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.
[20] 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.
[21] 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.
[22] Yinbo Li and Gonzalo R. Arce. A maximum likelihood approach to least absolute deviation regression. *EURASIP J. Appl. Signal Process.*, 2004:1762–1769, 2004.
[23] 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.
[24] Stphane Mallat and Zhifeng Zhang. Matching pursuit with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41:3397–3415, 1993.
[25] J.M. Mendel and C.S. Currus. *Maximum-likelihood deconvolution: a journey into model-based signal processing*. Springer-Verlag, 1990.
[26] Muhammad Aslam Noor. Splitting algorithms for general pseudomonotone mixed variational inequalities. *J. of Global Optimization*, 18(1):75–89, 2000.
[27] 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.
[28] T. Olofsson and E. Wennerstrom. Sparse deconvolution of b-scan images. *IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control*, 54:1634–1641, 2007.
[29] Tomas Olofsson. Semi-sparse deconvolution robust to uncertainties in the impulse responses. *Ultrasonics*, 37:423–432, 1999.

[30] Art B. Owen. A robust hybrid of lasso and ridge regression. Technical report, Stanford University, 2006.
[31] 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.
[32] K. R. Rao and P. Yip. *Discrete cosine transform: algorithms, advantages, applications*. Academic Press Professional, Inc., San Diego, CA, USA, 1990.
[33] Mark Rudelson and Roman Vershynin. Geometric approach to error correcting codes and reconstruction of signals. *Int. Math. Res. Not*, 64:4019–4041, 2005.
[34] OLOFSSON T. and STEPINSKI T. Maximum a posteriori deconvolution of sparse ultrasonic signals using genetic optimization. *Ultrasonics*, 37:423–432, 1999.
[35] V. N. Temlyakov. Greedy algorithms and m-term approximation with regard to redundant dictionaries. *J. Approx. Theory*, 98(1):117–145, 1999.
[36] V.N. Temlyakov. Nonlinear methods of approximation. *Found. Comput. Math.*, 3:33–107, 2002.
[37] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.
[38] Robert Tibshirani, Michael Saunders, Saharon Rosset, Ji Zhu, and Keith Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society Series B*, 67(1):91–108, 2005.
[39] Robert Tibshirani and Pei Wang. Spatial smoothing and hot spot detection for cgh data using the fused lasso. *Biostatistics (Oxford, England)*, 9(1):18–29, January 2008.
[40] J.A. Tropp. Just relax: convex programming methods for identifying sparse signals in noise. *Information Theory, IEEE Transactions on*, 52:1030–1051, 2006.
[41] 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.
[42] Joel A. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Trans. Inform. Theory*, 50:2231–2242, 2004.
[43] P. Tseng. Prediction accuracy and stability of regression with optimal scaling transformations. *Technical report LIDS-P;1840, Massachusetts Institute of Technology. Laboratory for Information and Decision Systems*, 1988.
[44] P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *J. Optim. Theory Appl.*, 109(3):475–494, 2001.
[45] Paul Tseng. A modified forward-backward splitting method for maximal monotone mappings. *SIAM J. Control Optim*, 38:431–446, 1998.
[46] A. Van der Kooij. Prediction accuracy and stability of regression with optimal scaling transformations. *Technical report, Dept. of Data Theory, Leiden University*, 2007.
[47] Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67, 2006.
[48] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society B*, 67:301–320, 2005.