Robust Sparse Coding and Compressed Sensing with the Difference Map

Will Landecker[1,2,⋆], Rick Chartrand[3], and Simon DeDeo[2,4]

[1] Portland State University, USA
[2] Santa Fe Institute, USA
[3] Los Alamos National Laboratory, USA
[4] Indiana University, USA

Abstract. In compressed sensing, we wish to reconstruct a sparse signal x from observed data y. In sparse coding, on the other hand, we wish to find a representation of an observed signal y as a sparse linear combination, with coefficients x, of elements from an overcomplete dictionary. While many algorithms are competitive at both problems when x is very sparse, it can be challenging to recover x when it is *less* sparse. We present the *Difference Map*, which excels at sparse recovery when sparseness is lower. The Difference Map out-performs the state of the art with reconstruction from random measurements and natural image reconstruction via sparse coding.

Keywords: Sparse coding, compressed sensing.

1 Introduction

In compressed sensing (CS), we are given a measurement matrix $\Phi \in \mathbb{R}^{m\times n}$ (where $m < n$), observed data $y \in \mathbb{R}^m$, and we wish to recover a *sparse* $x \in \mathbb{R}^n$ such that

$$y = \Phi x.$$

The compressed sensing problem can then be written as

$$\arg\min_x \|x\|_0 \text{ subject to } y = \Phi x, \tag{1}$$

where $\|\cdot\|_0$ is the ℓ^0 penalty function, giving the number of nonzero elements. In the noisy case, where

$$\tilde{y} = \Phi x + \epsilon \cdot \mathcal{N}(0, 1)$$

is assumed to be a noisy observation, we often replace the linear constraints with quadratic ones:

$$\arg\min_x \|x\|_0 \text{ subject to } \|\tilde{y} - \Phi x\|_2^2 \le \delta \tag{2}$$

for some $\delta > 0$. The problem (2) can also be used for sparse coding (SC); in this setting, Φ is an overcomplete dictionary, y is a signal (such as an image patch), and we seek a sparse coefficient vector x.

⋆ Corresponding author: `landeckw@cecs.pdx.edu`

Recently, a variety of algorithms have achieved good results for a variety of CS and SC problems, including Least Angle Regression (LARS) [1], Iterative Soft Thresholding and its variants [2,3], Subspace Pursuit [4], Matching Pursuit and its variants [5,6], Iterative Hard Thresholding (IHT) and its variants [7,8,9], Iteratively Reweighted Least Squares (IRLS) [10], and the Alternating Direction Method of Multipliers (ADMM) [11,12].

Because solving problems (1) and (2) directly is known to be NP-hard [13], some approaches relax the ℓ^0 penalty to the convex ℓ^1 norm [14,1], while some attempt to address the ℓ^0 case directly [4,6,7], and still others consider any number of ℓ^p (quasi-)norms for $0 < p \leq 1$ [10,11,12]. In general, the challenge for CS and SC algorithms is to balance two competing constraints on the solution x^*: to accurately reconstruct the observed data y, and to be sparse.

This paper presents a method for solving CS and SC problems without relaxing the ℓ^0 constraint, using a general method known as the Difference Map [15]. The Difference Map (DM) has been used to solve a wide variety of constraint-intersection problems. Given sets A and B and distance-minimizing projections P_A and P_B[1], respectively, DM searches for a point $x^* \in A \cap B$. One iteration of DM is defined by $x \leftarrow D(x)$, where

$$D(x) = x + \beta \left[P_A \circ f_B(x) - P_B \circ f_A(x) \right], \tag{3}$$

in which

$$f_A(x) = P_A(x) - \left(P_A(x) - x \right) / \beta,$$
$$f_B(x) = P_B(x) + \left(P_B(x) - x \right) / \beta,$$

and $\beta \neq 0$. One can test for convergence by monitoring the value $\|P_A \circ f_B(x) - P_B \circ f_A(x)\|_2$, which vanishes when a solution is found. Recently, DM has achieved state-of-the-art performance on a variety of NP-hard nonconvex optimization problems including protein folding, k-SAT, and packing problems [15].

The rest of this paper is organized as follows. In Section 2, we introduce an adaptation of the Difference Map for compressed sensing and sparse coding, which we compare at a high level to other well-known algorithms. In Section 3, we compare the algorithms on CS problems using random measurements, and we reconstruct natural images via SC in Section 4.

2 Compressed Sensing and Sparse Coding with the Difference Map

Given a matrix $\Phi \in \mathbb{R}^{m \times n}$ (where $m < n$) and data $y \in \mathbb{R}^m$, we wish to find a sparse vector $x^* \in \mathbb{R}^n$ that is a solution to problem (2). We apply the Difference Map to this problem by defining the constraint sets

$$A = \{x \in \mathbb{R}^n : \|x\|_0 \leq s\},$$
$$B = \{x \in \mathbb{R}^n : \|\Phi x - y\|_2^2 \leq \delta\},$$

[1] By distance-minimizing projection, we mean that $P_A(x_0) = \arg\min_x \|x_0 - x\|_2$ subject to $x \in A$, and likewise for P_B.

for a pre-defined positive integer s and scalar $\delta > 0$.

The minimum-distance projection onto A is known as *hard thresholding*, and is defined by

$$P_A(x) = [x]_s, \tag{4}$$

where $[x]_s$ is obtained by setting to zero the $n - s$ components of x having the smallest absolute values.

A minimum-distance projection onto the set B involves solving a quadratically-constrained quadratic programming problem (QCQP), which can be very costly. We approximate this projection with:

$$P_B(x) = x - \Phi^+(\Phi x - y), \tag{5}$$

where $\Phi^+ = \Phi^T(\Phi\Phi^T)^{-1}$ is the Moore-Penrose pseudo-inverse of Φ.

The motivation for (5) comes from a simplification of our definition for B. Consider the set with linear constraints (as in (1)):

$$\{x \in \mathbb{R}^n : \Phi x = y\}.$$

The minimum distance projection onto this set is given by the linearly constrained quadratic programming (LCQP) problem

$$P(x_0) = \arg\min_{x\in\mathbb{R}^n} \tfrac{1}{2}\|x - x_0\|_2^2 \ \text{ such that } \ \Phi x = y. \tag{6}$$

The Lagrangian of this LCQP is

$$\mathcal{L}(x, \lambda) = \tfrac{1}{2}\|x - x_0\|_2^2 + \lambda(\Phi x - y).$$

The x that solves the LCQP (6) is a minimizer of $\mathcal{L}$, and is found by setting $\nabla_x(\mathcal{L}) = 0$, which yields

$$x = x_0 + \Phi^T\lambda. \tag{7}$$

Plugging (7) into $y = \Phi x$ and solving for λ gives

$$\lambda = (\Phi\Phi^T)^{-1}(\Phi x_0 - y).$$

Finally, we plug this into (7) to get

$$x = x_0 - \Phi^T(\Phi\Phi^T)^{-1}(\Phi x_0 - y),$$

as in (5).

Although the motivation for (5) comes from the assumption of non-noisy observations $y = \Phi x$, it in fact performs very well in the noisy case. In Figure 1, we see that the linearly-constrained P_B (LCQP) allows DM to converge much more quickly than the quadratically-constrained P_B (QCQP), even when given noisy observations. In this experiment, we chose a random $\Phi \in \mathbb{R}^{400\times1000}$ and $\|x\|_0 = 150$ (see Section 3 for details on constructing Φ and x). We calculated the noiseless $y = \Phi x$ and the noisy $\tilde{y} = y + \epsilon \cdot \mathcal{N}(0, 1)$ such that $\text{SNR}(y, \tilde{y}) = 20$ dB. The Difference Map was then given Φ and $\tilde{y}$, and asked to recover x using

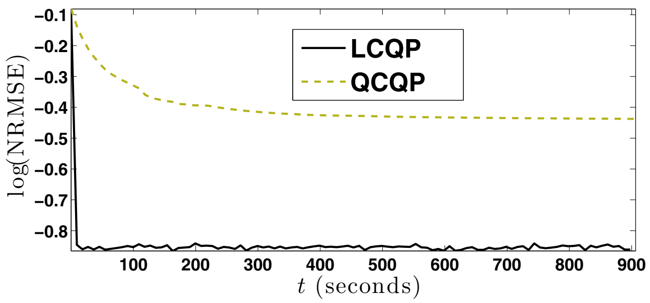


Fig. 1. The linearly constrained approximation (LCQP) of P_B allows the Difference Map to recover the signal x much more quickly than the quadratically constrained (QCQP) version of P_B, even when given the noisy observation $\tilde{y} = \Phi x + \epsilon \cdot \mathcal{N}(0, 1)$. We measure the log(NRMSE) of the estimate $\hat{x}$, computed by $\log(\|x - \hat{x}\|_2 / \|x\|_2)$. See text for additional details.

either the quadratically constrained P_B (QCQP) or the linearly constrained P_B (LCQP). The computationally expensive QCQP at each iteration causes DM to perform much more slowly. Thus we only consider the LCQP version of DM for the remainder of this work.

As stated above, the LCQP assumes that $\delta = 0$. Note that this assumption may reduce or even eliminate the intersection between the sets A and B. Nonetheless, Figure 1 shows that the speed improvement gained by this assumption far outweighs the disadvantages of the approximation, while still giving superior reconstruction for moderate noise (meaning $\delta > 0$).

It is worth noting that the pseudo-inverse is expensive to compute, though it only needs to be computed once. Thus in the case of sparse image reconstruction where each image patch is reconstructed independently, amortizing the cost of computing Φ^+ over all image patches significantly reduces the pre-computation overhead.

2.1 Comparison to other Algorithms

In what follows, we compare the Difference Map to a representative sample of commonly-used algorithms for solving CS and SC problems: Least Angle Regression (LARS) [1], Fast Iterative Soft Thresholding Algorithm (FISTA) [3], Stagewise Orthogonal Matching Pursuit (StOMP) [5], Accelerated Iterative Hard Thresholding (AIHT) [9], Subspace Pursuit (SP) [4], Iteratively-Reweighted Least Squares (IRLS) [10] and Alternating Direction Method of Multipliers (ADMM) [11,12]. As a final point of comparison, we test the Alternating Map (AM) defined by $x \leftarrow P_A(P_B(x))$, with P_A and P_B defined as in (4) and (5), respectively, which resembles the ECME algorithm for known sparsity levels [16].

The projection P_A (4), known as hard thresholding, is an important part of many CS algorithms [9,8,7,4,16,17]. The projection P_B (5) also appears in the ECME algorithm [16,17]. Normalized Iterative Hard Thresholding (NIHT) [8] uses a calculation similar to P_B, replacing the pseudo-inverse with $\mu_t \Phi^T$ for an appropriately chosen scalar μ_t.

Given that many algorithms consider the same types of projections as DM, any advantage achieved by DM must not come from the individual projections P_A and P_B, but rather the way in which DM combines the two projections into a single iterative procedure. This is particularly true when comparing DM to the simple alternating map. Alternating between projections is guaranteed to find a point at the intersection of the two constraints if both are convex; however, if either of the constraints is not convex, it is easy for this scheme to get stuck in a local minimum that does not belong to the intersection.

While many of the theoretical questions about the Difference Map remain open, it does come with a crucial guarantee here: even on nonconvex problems, a fixed point (meaning $D(x) = x$) implies that we have found a solution (meaning a point in $A \cap B$). To see this, note that $D(x) = x$ implies

$$P_A \circ f_B(x) = P_B \circ f_A(x). \tag{8}$$

Thus we have found a point that exists in both A and B. This leads us to believe that DM will find better sparse solutions when other algorithms are stuck in local minima.

Note that we are not the first to consider applying DM to compressed sensing. Qiu and Dogandžić [17] apply DM to the ECME algorithm (a variant of expectation maximization) in order to improve upon one of the two projections inside that algorithm. Although one of ECME's two projections uses DM *internally*, ECME continues to combine the two projections in a simple alternating fashion. This is in stark contrast to our proposed algorithm, which uses DM *externally* to the individual projections as a more intricate way of combining them. The resulting algorithm, called DM-ECME, is intended only for compressed sensing with non-negative signals. Because we consider different types of problems in this paper, we do not include DM-ECME in the comparisons below.

2.2 Implementation Details

We implemented the Difference Map in Matlab [18]. All experiments were performed on a computer with a 2.7 GHz quad-core Intel i7 processor, running Matlab R2013a. We obtained Matlab implementations of LARS and StOMP from SparseLab v2.1 [19]. Implementations of AIHT [9] and Subspace Pursuit [4] were found on the websites of the papers' authors. We also obtained Matlab implementations of ADMM [12] and IRLS [10] directly from the authors of the cited papers.

The implementations of LARS, Subspace Pursuit, AIHT, and StOMP are parameter-free. It was necessary to tune a single parameter for DM, and two parameters each for ADMM and IRLS. We tuned the parameters in two iterations of grid search. ADMM and IRLS required different parameters for the two different experiments presented in this paper (reconstruction from random measurements, and natural image reconstruction). Interestingly, DM performed well with the same parameter for both types of experiments.

We use "training" matrices of the same dimension, sparsity, and noise level as the ones presented in the figures of this paper in order to tune parameters. We chose parameters to minimize the MSE of the recovering x, averaged over all training problems. When tuning parameters for natural image reconstruction, we used a training set of 1000 image patches taken from the *person* and *hill* categories of ImageNet [20], providing a good variety of natural scenery.

When tuning β for DM, note that the values $\beta = 1$ and $\beta = -1$ greatly simplify the definitions of f_A and f_B and reduce the amount of computation per iteration by approximately one half. Thus these values are preferable when they lead to good performance. If they do not, one must resort to heuristics like grid search, as we do here. We first perform grid search with an interval of 0.1, between -1.2 and 1.2[2]. Next, in a radius of 0.05 around the best β, we performed another grid search with an interval of 0.01. Surprisingly, all β in the interval $[-0.9, -0.1]$ appeared to be equally good for all problems reported in this paper. We chose $\beta = -0.14$ because it performed slightly better during our experiments, but the advantage over other $\beta \in [-0.9, -0.1]$ was not significant.

We use logarithmic grid search to tune the two parameters for ADMM and IRLS. First, we search parameter values by powers of ten, meaning 10^{α}, for $\alpha = -5, -4, \ldots, 5$. We then search in the neighborhood of best exponent c by $\frac{1}{10}$ powers of ten, meaning $10^{c+\alpha}$ for $\alpha = -0.5, -0.4, \ldots, 0.5$.

For random measurements (the experiments in Section 3), this results in parameter values $\mu = 1.26 \times 10^2, \lambda = 3.98 \times 10^{-1}$ for ADMM, and $\alpha = 3.16 \times 10^{-3}; \beta = 2.51 \times 10^{-1}$. For natural image reconstruction (Section 4), we found $\mu = 1.58 \times 10^2, \lambda = 1.0 \times 10^{-1}$ for ADMM and $\alpha = 2.5 \times 10^{-4}, \beta = 5 \times 10^{-3}$ for IRLS. Note that the β parameter for IRLS has nothing to do with the β parameter for DM. We refer to both as β only to remain consistent with the respective bodies of literature about each algorithm, but in what follows we will only refer to the parameter for DM. IRLS is capable of addressing the ℓ^p quasi-norm for a variety of values $0 < p \leq 1$, while ADMM uses modifications of the ℓ^p quasi-norm designed to have a simple proximal mapping [21]. In both cases we tried $p = \frac{1}{2}$ and $p = 1$, and found $p = \frac{1}{2}$ to perform better.

3 Random Measurements

In this Section, we compare the performance of DM to other algorithms when reconstructing signals from random measurements, testing a wide variety of matrix sizes, sparsity, and noise levels. Given positive integers m, n, and s, we generate the random matrix $\Phi \in \mathbb{R}^{m \times n}$ with entries drawn from $\mathcal{N}(0, 1)$. We then ensure that columns have zero mean and unit norm. We generate the s-sparse vector $x \in \mathbb{R}^n$ whose nonzero elements are drawn from $\mathcal{N}(0, 1)$. We then calculate $y = \Phi x$, and the noisy "observation" $\tilde{y} = y + \epsilon \cdot \mathcal{N}(0, 1)$. Finally, we ask each algorithm to reconstruct x given only Φ and $\tilde{y}$.

[2] The natural range for the parameter β is [-1,1] (excluding 0), but Elser *et al.* [15] report that occasionally values outside of this interval work well.

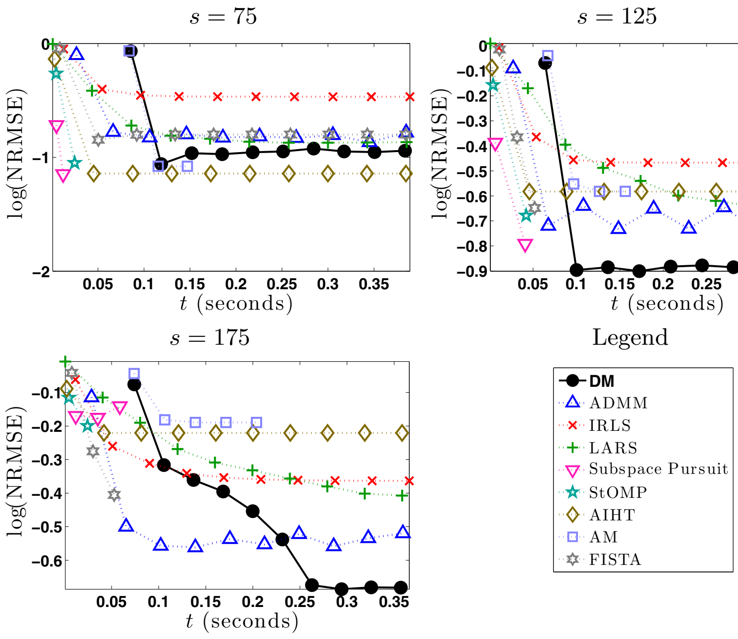


Fig. 2. Reconstructing signals with various levels of sparsity s. Given y and Φ, we try to recover x such that $y = \Phi x$ and $\|x\|_0 \leq s$. We measure the normalized root mean squared error (NRMSE) at time t by estimating x_t and calculating $\|x - x_t\|/\|x\|$. With sparser signals (upper left), most algorithms get equally close to recovering the true signal. With less sparse signals (upper right, lower left), the Difference Map gets closer than other algorithms to recovering the signal. Each plot is averaged over ten runs, with ϵ chosen to give an SNR of approximately 20 dB, and $\Phi \in \mathbb{R}^{400\times 1000}$.

We measure runtime instead of iterations, as the time required per iteration varies widely for the algorithms considered. Additionally, the pre-computation for DM is the longest of any algorithm, requiring the pseudo-inverse of the dictionary. This pre-computation overhead is included in the timekeeping.

In the first experiment, each algorithm attempts to reconstruct x as we vary the sparsity level s. We choose ϵ so that the signal-to-noise ratio (SNR) is close to 20 dB. The results in Figure 2 demonstrate that for small values of s (upper left), meaning sparser signals, most algorithms are able to recover x about equally well. As we increase the value of s (upper right, lower left), meaning denser signals, other algorithms converge to undesirable minima. The Difference Map, however, continues to get very close to recovering x.

In the next experiment, each algorithm attempts to reconstruct x as we vary the noise by changing ϵ. We fix s at 150. The results in Figure 3 show that with very little noise (upper left) and very high noise (lower right), the Difference Map performs as well as several algorithms at recovering the true signal x, though it requires more time. For moderate amounts of noise (upper right, lower left), the Difference Map is able to get closer to recovering the signal than any other algorithm.

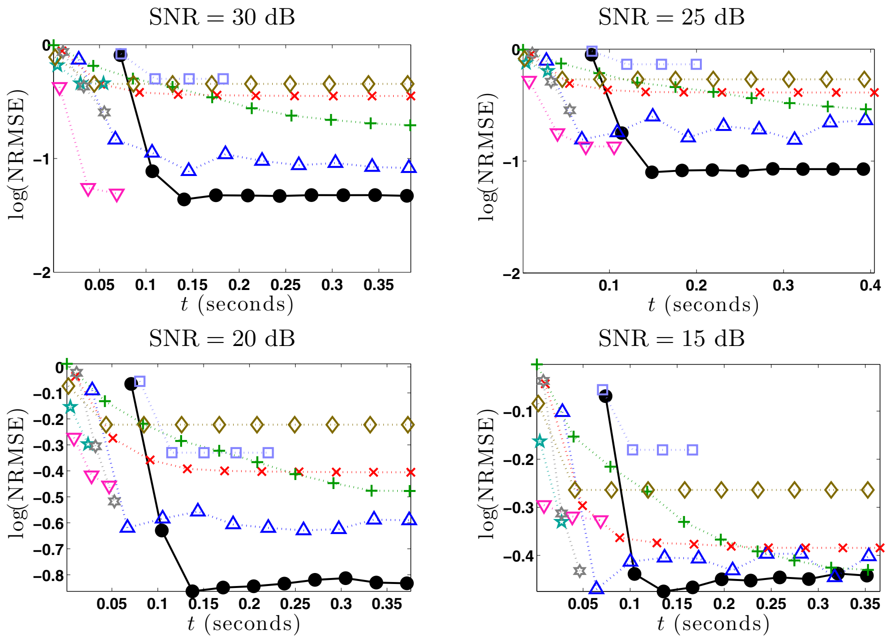


Fig. 3. Reconstructing signals with various levels of noise ϵ. Legend is the same as Figure 2. With very little noise (upper left) and large amounts of noise (bottom right), the Difference Map recovers the signal as well as the best algorithms, though requiring more time. With moderate amounts noise (upper right, lower left), the Difference Map gets closer than other algorithms to recovering the signal. Each plot is averaged over ten runs, with $s = 150$ and $\Phi \in \mathbb{R}^{400\times1000}$.

Note that DM and AM start "late" in all plots from Figures 2 and 3 because their pre-computation time is the longest (calculating Φ^+). Despite using the same projections, we notice a large disparity in performance between DM and AM when $s > 75$. Because the two algorithms both use the same two projections P_A and P_B, this performance gap shows the power of combining two simple projections in a more elaborate way than simply alternating between them.

From the results in Figures 2 and 3, we hypothesize that DM has a significant advantage with less sparse signals containing a moderate amount of noise, where other state-of-the-art compressed sensing algorithms get stuck in local minima or require a large amount of time to reach a good solution. We test this hypothesis with a variety of different matrix sizes and sparsity ratios in Figure 4, each time with an SNR of approximately 20 dB. The results show that DM does indeed outperform other algorithms in this setting, for all cases tested.

For all experiments reported above, the Difference Map's ℓ^0 constraint in (4) was the same as the true s used to generate the data. In many settings, however, the true s is unknown. We measure the robustness of DM in this setting by fixing the true value s (used to generate x) while varying the ℓ^0 constraint in (4). We then measure the log-NRMSE of the reconstructed signal $\hat{x}$. The results in Figure

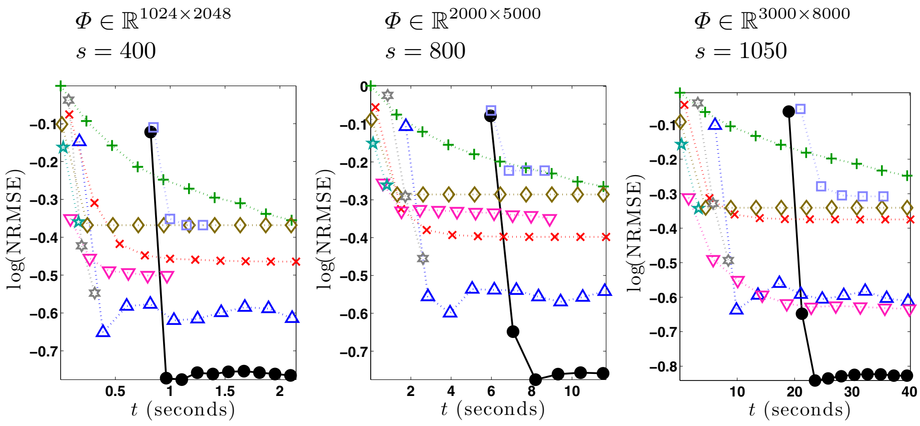


Fig. 4. In the low sparsity regime, the difference map outperforms other algorithms at recovering x from a noisy observed signal with a wide variety of matrix sizes $\mathbb{R}^{m\times n}$. Legend is the same as Figure 2. The noisy observation $\tilde{y} = \Phi x + \epsilon \cdot \mathcal{N}(0,1)$ has an SNR of approximately 20 dB. Each plot is averaged over ten runs.

5 show that when s is fixed at 150, DM continues to recover x better than any other algorithm for an ℓ^0 constraint down to 90 and up to 190. Thus DM appears quite robust to the specific ℓ^0 constraint value used when implementing the algorithm; we explore this property further when reconstructing natural images in Section 4, for which the "true s" is unknown.

4 Sparse Coding Image Reconstruction

The results from Section 3 indicate that DM offers an advantage over other algorithms when the underlying signal is *less* sparse and the observation is noisy. The less-sparse, noisy setting corresponds well to images which contain a large variety of textures, such as natural images. In this Section, we measure the sparse coding performance of the same algorithms as in the previous section (with the exception of AM, whose sparse coding performance was not competitive), by comparing reconstruction quality for a variety of images.

When reconstructing a large image, we treat each $w \times w$ patch as an independent signal to reconstruct. Because our dictionary is constant, we only need to compute the pseudo-inverse in (5) once. By amortizing the cost of the pseudo-inversion over all patches, this effectively allows DM to converge in less time per patch. We amortize the cost of pre-computation for other algorithms as well (most notably ADMM and IRLS).

In order to test the performance of the algorithms when reconstructing natural images, we require a dictionary learned for sparse image reconstruction. Dictionary learning is not the focus of this paper, but we present our method for completeness. The dictionary is trained with 10 million 20×20 image patches, and we choose to learn 1000 atoms, resulting in a dictionary of size 400×1000.

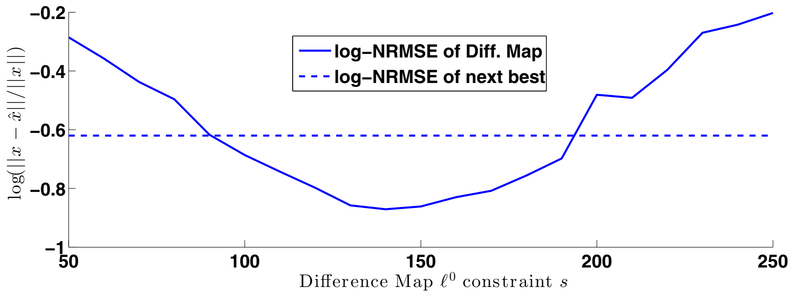


Fig. 5. The Difference Map outperforms other algorithms even when s is unknown, for a wide range of values. Using a random $\Phi \in \mathbb{R}^{400\times1000}$, $s = 150$, and ϵ chosen to give an SNR of 20 dB, we vary the Difference Map ℓ^0 constraint. The next best algorithm achieves a log-NRMSE of -0.62; the Difference Map outperforms this for any ℓ^0 constraint between 90 and 190.

We train the dictionary with patches from the *person* and *hills* category of ImageNet [20], which provide a variety of natural scenery. We alternate sparse coding using 20 iterations of ADMM using p-shrinkage with $p = 1/2$ (see [21] for details), with a dictionary update using the method of optimal directions [22]. We used ADMM as the sparse-coding algorithm simply because we had access to MPI-parallelized C code for this purpose. We do not believe that this gave an unfair advantage to ADMM, because the reconstructed images presented in this paper are separate from the dataset used to train the dictionary.

Using 1,000 processors, the dictionary converged in about 2.5 hours. The training patches were reconstructed by the dictionary with an average of 29 nonzero components (out of 1000), and the reconstruction of the training images had a relative error of 5.7%. The dictionary contains the typical combination of high- and low-frequency edges, at various orientations and scales. Some examples are shown in Figure 6.

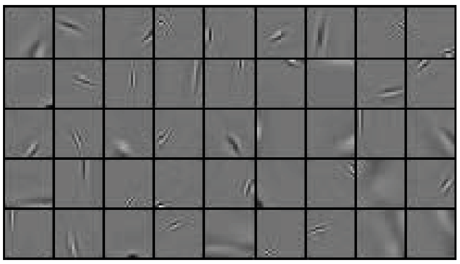


Fig. 6. Example atoms from the dictionary $\Phi \in \mathbb{R}^{400\times1000}$ used for reconstruction. The dictionary contains elements of size 20×20, learned from 10 million image patches from the *person* and *hill* categories of ImageNet [20].

Table 1. Signal to noise ratio (SNR, in decibels) of the reconstructed image from Figure 7. We test various sparsity levels s and various runtimes t (seconds per entire image). The difference map consistently achieves high SNR.

	$s = 100$			$s = 200$		
	$t = 10$	$t = 20$	$t = 30$	$t = 10$	$t = 20$	$t = 30$
Diff. Map	15.91	**17.55**	**17.45**	**20.28**	**22.38**	**23.34**
FISTA	4.80	12.04	17.21	4.82	12.13	21.71
ADMM	15.51	16.71	17.31	19.80	21.96	23.00
IRLS	9.62	13.52	14.92	13.47	18.38	20.62
Sub. Pursuit	**16.47**	16.78	16.84	16.94	16.88	16.87
LARS	10.62	12.66	14.16	10.63	12.68	14.24
AIHT	14.71	15.62	16.18	18.72	19.91	20.69
StOMP	15.55	15.50	15.50	17.65	17.92	17.93

Original	LARS	StOMP	IRLS	ADMM	SP	AIHT	FISTA	Diff. Map
	14.24 dB	17.93 dB	20.62 dB	23.00 dB	16.87 dB	20.69 dB	21.71 dB	**23.34 dB**

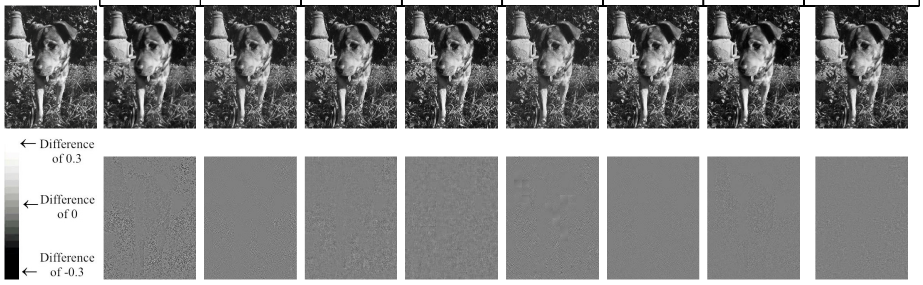


Fig. 7. Reconstructing a natural image. The Difference Map outperforms the other algorithms (SNR shown in decibels, top row) when reconstructing a 320×240 image of a dog (reconstructions shown in middle row). Difference images (bottom row) show the difference between the reconstruction and the original image, which ranges from -0.3 (black) to 0.3 (white) – original grayscale values are between 0 (black) and 1 (white). Results for $s = 200$ and $t = 30$.

We reconstruct several natural images and measure the quality of the sparse reconstruction as a function of time. At time t, we measure the reconstruction quality of patch y as follows. First, we perform hard-thresholding on the algorithm's current guess x_t, setting the $n - s$ smallest absolute values to zero, yielding the s-sparse vector $[x_t]_s$. We then calculate the reconstruction

$$y_t = \Phi[x_t]_s$$

and measure the SNR of y (the true image patch) to y_t. Thus we are measuring how well, at time t, the algorithm can create a *sparse* reconstruction of y. Note that algorithms returning a solution that is *sparser* than required will not be affected by the hard-thresholding step.

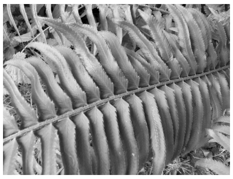
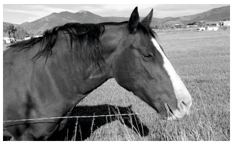
	Diff. Map	**23.19**		**Diff. Map**	**24.84**
	FISTA	19.49		FISTA	18.96
	ADMM	22.16		ADMM	23.78
	IRLS	20.58		IRLS	23.31
	LARS	14.51		LARS	18.02
	SP	17.85		SP	20.81
	StOMP	19.32		StOMP	23.23
	AIHT	20.37		AIHT	22.65
	Diff. Map	**22.79**		**Diff. Map**	**24.80**
	FISTA	19.00		FISTA	19.31
	ADMM	21.92		ADMM	24.06
	IRLS	20.51		IRLS	23.28
	LARS	14.81		LARS	18.00
	SP	17.89		SP	20.38
	StOMP	18.81		StOMP	21.13
	AIHT	20.08		AIHT	22.57

Fig. 8. The Difference Map regularly outperforms other algorithms in finding sparse reconstructions of a variety of images. We measure the SNR in decibels between the reconstruction and the original image (left column). Images are scaled to 240×320 pixels (320×240 for horizontal images). Reconstructions have sparsity $s = 200$, and are completed in 30 seconds per image (approximately 0.15 seconds per 20×20 patch). The dictionary Φ is the same as in Figure 6.

We reconstruct a 320×240 image of a dog, seen in Figure 7, using the 400×1000 dictionary from Figure 6. We measure results for both $s = 100$ and $s = 200$, as well as $t = 10, 20$ and 30 seconds[3]. The results in Table 1 show that DM consistently achieves a very good SNR of the reconstruction. As would be expected, increasing s and t tend to improve each algorithm's reconstruction performance.

The highest quality reconstructions, achieved with $s = 200$ and $t = 30$, are shown in Figure 7. While some algorithms fail to reconstruct details in the animal's fur and the grass, many algorithms reconstruct the image well enough to make it difficult to find errors by mere visual inspection. We show the difference between the reconstructions and the original image (Figure 7, bottom row), where a neutral gray color in the difference image corresponds to a perfect reconstruction of that pixel; white and black are scaled to a difference of 0.3 and -0.3, respectively (the original image was scaled to the interval [0,1]).

[3] We measure time in seconds per full-image reconstruction, which is actually performed independently for each 20×20 patch. Thus $t = 10, 20$ and 30 correspond to approximately $0.05, 0.1$, and 0.15 seconds per patch, respectively. The astute reader will notice that when Φ has dimension 400×1000, it takes longer than 0.05 seconds to compute Φ^+. This can be seen in Figure 3, where it takes almost 0.1 seconds for DM to finish calculating Φ^+ and begin searching for x. However, because we only need to calculate the pseudo-inverse once for the entire image, this start-up cost is amortized over all patches and becomes negligible.

The advantage of DM over other algorithms, when sparsely reconstructing images, can be seen with a large variety of images. In Figure 8, we see that DM consistently achieves the best reconstruction. All original images are included in the Supplementary Material.

5 Conclusions

We have presented the Difference Map, a method of finding a point at the intersection of two constraint sets, and we have introduced an implementation of DM for sparse coding and compressed sensing. The constraint-set formulation is a natural fit for sparse recovery problems, in which we have two competing constraints for x: to be consistent with the data y, and to be sparse.

When the solution x is very sparse and the observation $\tilde{y}$ is not too noisy, DM takes more time in finding the same solution as competing algorithms. However, when the solution x is *less* sparse and when the observation $\tilde{y}$ is noisy, DM outperforms state of the art sparse recovery algorithms. The noisy, less sparse setting corresponds well to reconstructing natural images, which can often require a large number of components in order to accurately reconstruct. Experiments show that DM performs favorably in reconstructing a variety of images, with a variety of parameter settings.

Parameter tuning can present a laborious hurdle to the researcher. DM requires tuning only a single parameter β. For all experiments in this paper (natural image reconstruction for various images; reconstruction with random matrix dictionaries of various sizes, with varying amounts of sparsity and noise), we found DM to work almost equally as well for all $-0.9 \leq \beta \leq -0.1$. The robustness of DM under such a wide variety of parameter values and problems makes DM a very competitive choice for compressed sensing.

In the case where s is unknown, it effectively becomes a separate parameter for many algorithms, including DM. However, we have shown in Figure 5 and in all experiments in Section 4 that DM maintains superior performance over competing algorithms even when s is unknown.

The robustness of DM comes from how it combines two simple projections into a single iterative procedure. The Alternating Map (AM) combines the same projections in a simple alternating fashion, and struggles in almost all experiments. The gap in performance between these two methods demonstrates the power of combining multiple constraints in a more perspicacious way.

Finally, we recall that performance in all experiments was measured as a function of time, which would seem to put DM at a natural disadvantage to other algorithms: DM requires the pseudo-inverse of the dictionary, computing which requires more time than any other algorithm's pre-computation. Despite this, DM consistently outperforms other algorithms.

Acknowledgements. SD and WL were supported by National Science Foundation Grant EF- 1137929, "The Small Number Limit of Biological Information Processing," and the Emergent Institutions Project.

RC was supported by the U.S. Department of Energy through the LANL/LDRD Program, and by the University of California Laboratory Fees Research Program.

References

1. Efron, B., Hastie, T., Johnstone, I., Tibshirani, R.: Least angle regression. Annals of Statistics 32, 407–499 (2004)
2. Daubechies, I., Defrise, M., De Mol, C.: An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. Communications in Pure and Applied Mathematics 57(11), 1413–1457 (2004)
3. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM Journal on Imaging Sciences 2(1), 183–202 (2009)
4. Dai, W., Milenkovic, O.: Subspace pursuit for compressive sensing signal reconstruction. IEEE Transactions on Information Theory 55(5), 2230–2249 (2009)
5. Donoho, D.L., Tsaig, Y., Drori, I., Starck, J.L.: Sparse solution of underdetermined systems of linear equations by stagewise orthogonal matching pursuit. IEEE Transactions on Information Theory 58(2), 1094–1121 (2012)
6. Tropp, J.A., Gilbert, A.C.: Signal recovery from random measurements via orthogonal matching pursuit. IEEE Transactions on Information Theory 53(12), 4655–4666 (2007)
7. Blumensath, T., Davies, M.E.: Normalized iterative hard thresholding: Guaranteed stability and performance. IEEE Journal of Selected Topics in Signal Processing, 298–309 (2010)
8. Blumensath, T., Davies, M.E.: Iterative hard thresholding for compressed sensing. Applied and Computational Harmonic Analysis 27(3), 265–274 (2009)
9. Blumensath, T.: Accelerated iterative hard thresholding. Signal Processing 92(3), 752–756 (2012)
10. Chartrand, R., Yin, W.: Iteratively reweighted algorithms for compressive sensing. In: IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 3869–3872 (2008)
11. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends in Machine Learning 3(1), 1–122 (2011)
12. Chartrand, R., Wohlberg, B.: A nonconvex ADMM algorithm for group sparsity with sparse groups. In: IEEE International Conference on Acoustics, Speech, and Signal Processing (2013)
13. Natarajan, B.K.: Sparse approximate solutions to linear systems. SIAM Journal on Computing 24(2), 227–234 (1995)
14. Tibshirani, R.: Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society. Series B (Methodological), 267–288 (1996)
15. Elser, V., Rankenburg, I., Thibault, P.: Searching with iterated maps. Proceedings of the National Academy of Sciences 104(2), 418–423 (2007)
16. Qiu, K., Dogandžić, A.: Double overrelaxation thresholding methods for sparse signal reconstruction. In: IEEE Information Sciences and Systems, pp. 1–6 (2010)
17. Qiu, K., Dogandžić, A.: Nonnegative signal reconstruction from compressive samples via a difference map ECME algorithm. In: IEEE Workshop on Statistical Signal Processing, pp. 561–564 (2011)
18. Landecker, W., Chartrand, R., DeDeo, S.: Matlab code for sparse coding and compressed sensing with the Difference Map, `http://santafe.edu/~simon/dm-cs0.zip`

19. Stodden, V., Carlin, L., Donoho, D., Drori, I., Dunson, D., Elad, M., Ji, S., Starck, J., Tanner, J., Temlyakov, V., Tsaig, Y., Xue, Y.: SparseLab. Matlab software package,
`http://sparselab.stanford.edu/`
20. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: A Large-Scale Hierarchical Image Database. In: IEEE Conference on Computer Vision and Pattern Recognition (2009)
21. Chartrand, R.: Nonconvex splitting for regularized low-rank + sparse decomposition. IEEE Transactions on Signal Processing 60, 5810–5819 (2012)
22. Engan, K., Aase, S., Hakon Husoy, J.: Method of optimal directions for frame design. In: IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 2443–2446 (1999)