HOMOTOPY CONTINUATION FOR SPARSE SIGNAL REPRESENTATION

Dmitry M. Malioutov, Müjdat Çetin, and Alan S. Willsky

Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
77 Massachusetts Ave., Cambridge, MA 02139, USA

ABSTRACT

We explore the application of a homotopy continuation-based method for sparse signal representation in overcomplete dictionaries. Our problem setup is based on the basis pursuit framework, which involves a convex optimization problem consisting of terms enforcing data fidelity and sparsity, balanced by a regularization parameter. Choosing a good regularization parameter in this framework is a challenging task. We describe a homotopy continuation-based algorithm to efficiently find and trace all solutions of basis pursuit as a function of the regularization parameter. In addition to providing an attractive alternative to existing optimization methods for solving the basis pursuit problem, this algorithm can also be used to provide an automatic choice for the regularization parameter, based on prior information about the desired number of non-zero components in the sparse representation. Our numerical examples demonstrate the effectiveness of this algorithm in accurately and efficiently generating entire solution paths for basis pursuit, as well as producing reasonable regularization parameter choices. Furthermore, exploring the resulting solution paths in various operating conditions reveals insights about the nature of basis pursuit solutions.

1. INTRODUCTION

Representing data in the most parsimonious fashion in terms of redundant collections of generating elements is at the core of many signal processing applications. However, finding such sparse representations exactly in terms of overcomplete dictionaries involves the solution of intractable combinatorial optimization problems. As a result, work in this area has focused on approximate methods, based on convex relaxations [1] or greedy methods, leading recently to the development of conditions under which such methods yield maximally sparse representations [2–6]. One such method, involving a convex ℓ_1 relaxation, is basis pursuit [1]. Its noisy version (allowing for some residual mismatch to data) poses the following optimization problem:

$$J(\mathbf{x};\lambda) = \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_1, \ \ \mathbf{A} \in \mathbb{R}^{M\times N} \qquad (1)$$

where $\mathbf{y}$ denotes the data (signal whose representation we seek), $\mathbf{A}$ is the overcomplete representation dictionary ($M < N$), and $\lambda \geq 0$ is a scalar regularization parameter, balancing the tradeoff between sparsity and residual error. For a fixed λ, the problem can be solved by finding the minimizer $\hat{\mathbf{x}}$ of (1), using e.g. quadratic programming. However choosing the regularization parameter is a difficult task, and some prior knowledge, either of the desired residual error (e.g. based on the noise level), or of the underlying sparse vector $\mathbf{x}$, has to be exploited. One piece of information about $\mathbf{x}$ might be the number of non-zero components. However, even if such information is available, how to use it directly in the basis pursuit framework is not straightforward.

Motivated by these observations, we describe a computationally efficient approach for sparse signal representation based on the homotopy continuation method of [7]. A related method has also been developed in [8], and has been linked to greedy methods. The main focus in [7] is the solution of an overdetermined least-squares problem with an ℓ_1-norm constraint. We are mostly interested in the unconstrained formulation in (1), in the underdetermined ($M < N$) case. In particular, we propose a simple algorithm to find and trace all solutions $\hat{\mathbf{x}}(\lambda)$ of basis pursuit as a function of the regularization parameter λ. The function $J(\mathbf{x};\lambda)$ is convex and hence continuous, but it is not differentiable whenever $x_i = 0$ for some i, due to the term $\|\mathbf{x}\|_1 = \sum_i |x_i|$. The main idea of the approach is that $\|\mathbf{x}\|_1$, when restricted to the subset of non-zero indices of $\mathbf{x}$, is locally a linear function of $\mathbf{x}$. This allows one to solve the local problems (for a limited range of λ) analytically, and piece together local solutions to get solutions for all regions of λ. The resulting algorithm generates solutions for all λ with a computational cost that is comparable to solving basis pursuit with quadratic programming for a single λ. This procedure can also be used to select the regularization parameter λ based on information about the number of non-zero components in $\mathbf{x}$. In particular, a reasonable choice is the minimum λ that produces the desired number of non-zero components in $\hat{\mathbf{x}}(\lambda)$. Our numerical experiments demonstrate the effectiveness of this algorithm in generating the solution path accurately. Furthermore, exploring the structure of such solution paths reveals useful insights about the sensitivity of the problem to measurement noise, as well as to the nature of the overcomplete dictionary used.

2. NON-SMOOTH OPTIMALITY CONDITIONS

First we review non-smooth optimality conditions for convex functions and their implications for the problem in (1).

The subdifferential of a convex function $f : \mathbb{R}^N \to \mathbb{R}$ at $\mathbf{x} \in \mathbb{R}^N$ is defined as the following set:

$$\partial f(\mathbf{x}) = \{\xi \in \mathbb{R}^N | f(\mathbf{y}) \geq f(\mathbf{x}) + \xi^T(\mathbf{y} - \mathbf{x}) \ \forall\, \mathbf{y} \in \mathbb{R}^N\} \quad (2)$$

Each element of $\partial f(\mathbf{x})$ is called a subgradient of f at $\mathbf{x}$. The subdifferential is a generalization of the gradient of f. In fact, if f

This work was supported by the Army Research Office under Grant DAAD19-00-1-0466.

is convex *and* differentiable at a point $\mathbf{x}$ then

$$\partial f(\mathbf{x}) = \{\nabla f(\mathbf{x})\} \tag{3}$$

i.e. the subdifferential consists of a single vector, the gradient of f at $\mathbf{x}$ (the only subgradient is the gradient).

The non-smooth optimality conditions state that the subdifferential of f at $\mathbf{x}$ has to contain the $\mathbf{0}$-vector for f to achieve a global minimum at $\mathbf{x}$:

Theorem 1 (Non-smooth optimality conditions) *If $f : \mathbb{R}^N \to \mathbb{R}$ is convex, then f attains a global minimum at $\mathbf{x}$ if and only if $\mathbf{0} \in \partial f(\mathbf{x})$.*

The subdifferential of $g(\mathbf{x}) = \|\mathbf{x}\|_1$ is the following set:

$$\mathbf{u}(\mathbf{x}) \triangleq \partial g = \left\{ \mathbf{u} \in \mathbb{R}^N \;\middle|\; \begin{array}{ll} u_i = 1 & \text{if } x_i > 0 \\ u_i = -1 & \text{if } x_i < 0 \\ u_i \in [-1, .., 1] & \text{if } x_i = 0 \end{array} \right\} \tag{4}$$

The interesting part of this subdifferential is when some of the coordinates are equal to 0, where g is non-differentiable. Then u_i is not a scalar, it is a set.

The subdifferential of $f(\mathbf{x}) = J(\mathbf{x}; \lambda)$ from (1), for a fixed $\lambda = \tilde{\lambda}$, is the set

$$\partial f = \left\{ 2\mathbf{A}'(\mathbf{A}\mathbf{x} - \mathbf{y}) + \tilde{\lambda}\mathbf{u}(\mathbf{x}) \right\} \tag{5}$$

where $\mathbf{u}(\mathbf{x})$ is defined above in (4). Suppose that $\tilde{\mathbf{x}} = \arg\min_{\mathbf{x}} J(\mathbf{x}; \tilde{\lambda})$. Then, in order to have $\mathbf{0} \in \partial f(\tilde{\mathbf{x}})$, the following equation must have a solution for some vector $\tilde{\mathbf{u}} \in \mathbf{u}(\tilde{\mathbf{x}})$:

$$2\mathbf{A}'\mathbf{A}\tilde{\mathbf{x}} + \tilde{\lambda}\tilde{\mathbf{u}} = 2\mathbf{A}'\mathbf{y} \tag{6}$$

Let us consider an arbitrary vector $\mathbf{x}$ more closely. Let $\mathcal{I}_{on}$ be the support of $\mathbf{x}$, i.e. the set of indices i where $x_i \neq 0$. Also let $\mathcal{I}_{off}$ be the complement of $\mathcal{I}_{on}$, i.e. $\mathcal{I}_{off} = \{i \mid x_i = 0\}$. Put all entries x_i on the support of $\mathbf{x}$ into a vector $\mathbf{x}_{on}$, and the ones off the support of $\mathbf{x}$ into $\mathbf{x}_{off}$ (that makes $\mathbf{x}_{off} = \mathbf{0}$). Assume, without loss of generality, that $\mathbf{x}' = [\mathbf{x}'_{on}\ ,\ \mathbf{x}'_{off}]$, i.e. the non-zero components appear first. Let us split $\mathbf{u}$ in the same fashion, according to which indices lie on or off the support of $\mathbf{x}$, into $\mathbf{u}_{on}$ and $\mathbf{u}_{off}$. Also, let us split the square $N \times N$ matrix $\mathbf{G} = 2\mathbf{A}'\mathbf{A}$ into 4 parts (there are 4 possibilities of whether the row-index and the column-index correspond to our sets $\mathcal{I}_{on}$ and $\mathcal{I}_{off}$): $\mathbf{G}_{on,on}$, $\mathbf{G}_{on,off}$, $\mathbf{G}_{off,on}$, $\mathbf{G}_{off,off}$. Due to symmetry of the matrix $\mathbf{G}$, we have $\mathbf{G}_{on,off} = \mathbf{G}'_{off,on}$. To simplify the notation further, let us use $\Phi = \mathbf{G}_{on,on}$, $\Psi = \mathbf{G}_{on,off}$, and $\Upsilon = \mathbf{G}_{off,off}$. Finally, let $\mathbf{z} = 2\mathbf{A}'\mathbf{y}$, and split $\mathbf{z}$ in the same way into $\mathbf{z}_{on}$ and $\mathbf{z}_{off}$.

Returning to our fixed $\tilde{\mathbf{x}}$ and $\tilde{\lambda}$, using our new notation, we can rewrite (6) as

$$\begin{pmatrix} \Phi & \Psi \\ \Psi' & \Upsilon \end{pmatrix} \begin{pmatrix} \tilde{\mathbf{x}}_{on} \\ \mathbf{0} \end{pmatrix} + \tilde{\lambda} \begin{pmatrix} \tilde{\mathbf{u}}_{on} \\ \tilde{\mathbf{u}}_{off} \end{pmatrix} = \begin{pmatrix} \mathbf{z}_{on} \\ \mathbf{z}_{off} \end{pmatrix} \tag{7}$$

Suppose that we know $\tilde{\mathbf{x}}$. The elements of $\tilde{\mathbf{u}}_{on}$ are all determined: they are equal to 1 or -1, corresponding to the signs of elements of $\tilde{\mathbf{x}}_{on}$. To determine $\tilde{\mathbf{u}}_{off}$, split equation (7) into two parts to get:

$$\begin{aligned} \Phi\tilde{\mathbf{x}}_{on} + \tilde{\lambda}\tilde{\mathbf{u}}_{on} &= \mathbf{z}_{on} \\ \Psi'\tilde{\mathbf{x}}_{on} + \tilde{\lambda}\tilde{\mathbf{u}}_{off} &= \mathbf{z}_{off} \end{aligned} \tag{8}$$

Thus we can find $\tilde{\mathbf{u}}_{off} = \frac{1}{\tilde{\lambda}}(\mathbf{z}_{off} - \Psi'\tilde{\mathbf{x}}_{on})$. Since $\tilde{\mathbf{x}}$ is optimal (for some $\lambda = \tilde{\lambda}$), the elements of $\tilde{\mathbf{u}}_{off}$ are constrained to lie in $[-1, 1]$.

3. FINDING SOLUTIONS FOR ALL λ

In the last section we characterized $\tilde{\mathbf{u}}$ given that we know $\tilde{\mathbf{x}}$, the optimal solution for a particular $\tilde{\lambda}$. Now starting with $\lambda = \tilde{\lambda}$, we incrementally change λ to find and trace optimal solutions $\hat{\mathbf{x}}(\lambda)$ for all λ. This forms the basis of the homotopy continuation method.

Suppose that $\tilde{\mathbf{x}}$ is the unique solution for $\tilde{\lambda}$ (where $\tilde{\lambda} > 0$), then from (8) we have[1]

$$\tilde{\mathbf{x}}_{on} = \Phi^{-1}(\mathbf{z}_{on} - \tilde{\lambda}\tilde{\mathbf{u}}_{on}) \tag{9}$$

$$\tilde{\mathbf{u}}_{off} = \frac{1}{\tilde{\lambda}}(\mathbf{z}_{off} - \Psi'\Phi^{-1}\mathbf{z}_{on}) + \Psi'\Phi^{-1}\tilde{\mathbf{u}}_{on} \tag{10}$$

No elements of $\tilde{\mathbf{x}}_{on}$ are equal to zero, hence there exists a range of λ, which includes $\tilde{\lambda}$, for which all entries of $\mathbf{x}_{on}(\lambda) = \Phi^{-1}(\mathbf{z}_{on} - \lambda\tilde{\mathbf{u}}_{on})$ will be nonzero. That means that throughout this range the support of $\mathbf{x}(\lambda)$ will not be reduced. By larger changes in λ we can force one of the components of $\mathbf{x}_{on}(\lambda)$ to zero. In addition, there exists a range of λ, which includes $\tilde{\lambda}$, for which $\mathbf{u}_{off}(\lambda) = \frac{1}{\lambda}(\mathbf{z}_{off} - \Psi'\Phi^{-1}\mathbf{z}_{on}) + \Psi'\Phi^{-1}\tilde{\mathbf{u}}_{on}$ does not become equal to 1 in absolute value, i.e. all entries of $\mathbf{u}_{off}(\lambda)$ belong to $[-1, 1]$. In the intersection of these two ranges of λ, the vectors $\mathbf{x}(\lambda)$ and $\mathbf{u}(\lambda)$ will satisfy the non-smooth optimality conditions for $J(\mathbf{x}(\lambda); \lambda)$, hence $\hat{\mathbf{x}}(\lambda) = \mathbf{x}(\lambda)$ for λ in the above region. The vector $\mathbf{x}(\lambda)$ is obtained by putting entries of $\mathbf{x}_{on}(\lambda)$ into the corresponding entries $\hat{x}_i(\lambda)$, for $i \in \mathcal{I}_{on}$, and zeros for $i \in \mathcal{I}_{off}$. The vector $\mathbf{u}(\lambda)$ is obtained by putting $\tilde{\mathbf{u}}_{on}$ (which does not change while λ is in the above region) into the components with $i \in \mathcal{I}_{on}$, and $\mathbf{u}_{off}(\lambda)$ for $i \in \mathcal{I}_{off}$.

In this way, we obtain all solutions for some range of λ's. The range can be easily calculated by solving for critical values of λ closest to $\tilde{\lambda}$, which make an entry of $\hat{\mathbf{x}}_{on}(\lambda)$ turn zero, or an entry of $\mathbf{u}_{off}(\lambda)$ reach unity in absolute value. This requires solving a set of scalar linear equations.

Now the next step is to find the support of $\hat{\mathbf{x}}(\lambda)$, as λ leaves the region. We only need to search locally, since $\hat{\mathbf{x}}(\lambda)$ is continuous for $\lambda > 0$ [7]. For the case where changing λ forces one component of $\mathbf{x}_{on}(\lambda)$ to zero, recalculating the support is trivial: we remove the index i for which x_i was set to zero from $\mathcal{I}_{on}$, and put it into $\mathcal{I}_{off}$. For the case where an entry of $\mathbf{u}_{off}(\lambda)$ becomes equal to 1 in absolute value, we transfer the corresponding index i from $\mathcal{I}_{off}$ into $\mathcal{I}_{on}$. The corresponding index of $\mathbf{u}_{on}$ is set to the sign of the entry of $\mathbf{u}_{off}(\lambda)$ which reached 1 in absolute value. Thus, after recomputing the support and the sign-pattern of solutions, we can proceed in the same fashion as before, computing the boundary of the new region for λ, finding the optimal solutions inside it, and entering a new region.

To start the algorithm, it is easiest[2] to consider $\lambda_0 = \infty$, or equivalently $\lambda_0 = 2\|\mathbf{A}'\mathbf{y}\|_\infty$, which satisfies $\hat{\mathbf{x}}(\lambda) = \mathbf{0}$ for $\lambda >$

[1] In this case, it can be shown that the matrix Φ is invertible.

[2] Another possibility is to start with $\lambda_0 = 0$, and increase it until $\hat{\mathbf{x}}(\lambda)$ becomes $\mathbf{0}$. Assuming that $\mathbf{A}$ has full row rank, this starting point requires the solution of the problem: $\min \|\mathbf{x}\|_1$ subject to $\mathbf{y} = \mathbf{A}\mathbf{x}$. The solution corresponds to $\lambda = 0^+$. When $\lambda = 0$ there exist multiple solutions if $\mathbf{A}$ has a nontrivial null-space. Solving the linear program picks the sparsest solution, which lies on the solution path $\hat{\mathbf{x}}(\lambda)$.

λ_0. Then, following the procedure described above, the algorithm produces $\hat{\mathbf{x}}(\lambda)$ for all $\lambda \geq 0$, and terminates when λ reaches 0.

The algorithm can exploit prior information about the desired number of non-zero elements in the representation to produce an automatic choice for the regularization parameter λ for basis pursuit. In particular, among all λ for which $\hat{\mathbf{x}}(\lambda)$ has the desired sparsity, the smallest one can be a reasonable choice in many scenarios, as it leads to the smallest residual, $\|\mathbf{y} - \mathbf{A}\hat{\mathbf{x}}(\lambda)\|_2$. One might also consider other choices for λ, guided by the structure of the solution path, as we discuss in Section 4.

The computational complexity of the algorithm is dominated by the inversion of the matrix Φ at each breakpoint, which is bounded by $O(M^3)$, where M is the number of rows of $\mathbf{A}$. However, at each breakpoint the rank of the matrix Φ is changed by adding (or removing) a row and a column, hence instead of computing the inverse from scratch, rank-one updates can be done at the cost of $O(M^2)$. Empirically, the number of breakpoints is around M, but more careful analysis is in order. Thus, the cost of finding the whole solution path is roughly the same as for one iteration of the Newton's method to solve the problem in (1) for a fixed λ, i.e. $O(M^3)$. In addition, if one does not need the full solution path $\hat{\mathbf{x}}(\lambda)$, but only the path from $\hat{\mathbf{x}}(\lambda_0) = \mathbf{0}$ to a solution with L components, then the complexity is bounded by $O(L^3)$, with L instead of M, and the number of breakpoints is typically around L. Thus, the method is extremely efficient in computing very sparse solutions starting from $\hat{\mathbf{x}}(\lambda_0) = \mathbf{0}$.

To conclude the section, let us comment on the numerical stability of the algorithm. When we switch from one region to another, the only information that is carried over is the support of the new optimal solution, and the signs. Hence, if a small numerical error due to finite precision is made in computing the optimal solution for one region of λ (small enough not to affect the support and signs of the solution at the region boundary), then in the next region this error has no effect at all. Thus, the algorithm has a self-stabilizing property.

4. NUMERICAL EXAMPLES

4.1. Small Analytical Example

First we consider a very small example with $\mathbf{A} \in \mathbb{R}^{2\times 3}$:

$$\mathbf{A} = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 1.5 \end{pmatrix}, \text{ and } \mathbf{y} = \begin{pmatrix} 6 \\ 6 \end{pmatrix}$$

We apply the algorithm from Section 3, and the resulting solution path is shown in Figure 1. For this small problem, we are also able to compute the entire solution path analytically, and observe that the algorithm produces it accurately. The two triangles are the intersections of $\mathbb{R}^+$ with the planes $x_1 + 2x_2 + 3x_3 = 6$, and $x_1 + 3x_2 + 1.5x_3 = 6$. The solution path $\hat{\mathbf{x}}(\lambda)$ starts at $\lambda = 60$, with $\mathbf{x} = \mathbf{0}$. As λ starts to decrease, the solution path enters a segment with one non-zero component: $x_2 = \frac{30}{13} - \frac{\lambda}{26}$, and $x_1 = x_3 = 0$. The segment satisfies optimality conditions until $\lambda = 28.8$, after which x_3 becomes non-zero. The solution path from $\lambda = 28.8$, down to $\lambda = 0^+$ is $x_2 = \frac{3}{2} - \frac{\lambda}{96}$, $x_3 = 1 - \frac{5\lambda}{144}$, and $x_1 = 0$. The minimum-norm solution, corresponding to $\lambda = 0$, is $\hat{\mathbf{x}}_{MN} = [.4968, 1.3758, .9172]$, is not sparse.

4.2. Larger Numerical Examples

Now we demonstrate the application of the algorithm on larger examples. We consider a problem $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{n}$, where $\mathbf{A}$ is an

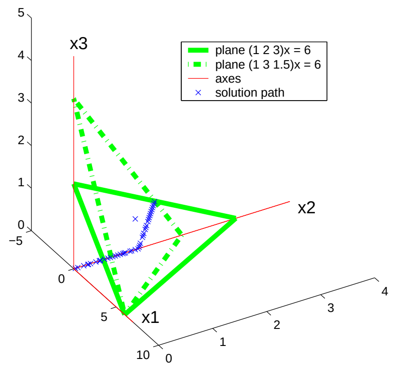

Fig. 1. Solution path for a small problem.

overcomplete 20×100 discrete cosine transform (DCT) dictionary, and $\mathbf{n}$ is zero-mean Gaussian noise. Dictionaries of this type arise naturally in many signal processing applications, one example being source localization with sensor arrays, where the observation model for linear arrays involves a discrete Fourier transform (DFT) dictionary [9]. In the specific example we consider here, $\mathbf{x}$ has two non-zero components, both equal to 1. In Figure 2 (top) we plot the solution path for noiseless data ($\mathbf{n} = \mathbf{0}$), in the middle plot for small amounts of noise (SNR = 15 dB), and in the bottom plot for moderate amounts of noise (SNR = 5 dB). Each piecewise-linear curve in these plots corresponds to one component $\hat{x}_i(\lambda)$. We also evaluate the solution at three intermediate values of λ in each linear segment, and compare it to a solution of the corresponding optimization problem in (1) using quadratic programming. The solutions agree almost perfectly, up to negligible numerical errors for all the examples.

Consider the top plot of Figure 2 which depicts the noiseless scenario. The smallest λ which leads to two non-zero components is $\lambda = 0^+$, which is the best parameter choice in this case. The corresponding solution found by homotopy-continuation has two non-zero entries equal to 1, and agrees with the original signal $\mathbf{x}$. In the middle plot, where the data are slightly noisy, the solution path ends at a non-sparse vector, which is close to the optimal solution of the noiseless problem (i.e. the other non-zero components are small). The smallest λ yielding exactly two non-zero components is $\lambda = 1.4548$. We note that the corresponding solution has non-zero indices not exactly equal, but very close to the ones of $\mathbf{x}$. The solution path suggests that an alternative to this choice of λ is to to pick a non-sparse solution for $\lambda = 0^+$ and threshold it, which would recover the exact indices in this mildly noisy scenario. In the bottom plot, the noise is sufficient to substantially change the solution path, but the smallest λ which leads to two non-zero elements ($\lambda = 0.6526$) still produces a reasonable solution, which is depicted in Figure 3 (we plot all components of $\hat{x}_i(\lambda)$ vs. i). Note that the indices of non-zero elements of $\hat{\mathbf{x}}(\lambda)$ are very close to those of the true $\mathbf{x}$. This 'stability' of indices of non-zero components occurs due to the special structure of $\mathbf{A}$: nearby columns of $\mathbf{A}$ are almost parallel for our overcomplete DCT matrix $\mathbf{A}$, and columns which are far apart are nearly orthogonal. This structure is what allows sparse signal representation ideas to be applied to source localization-type problems, even for highly overcomplete dictionaries [9].

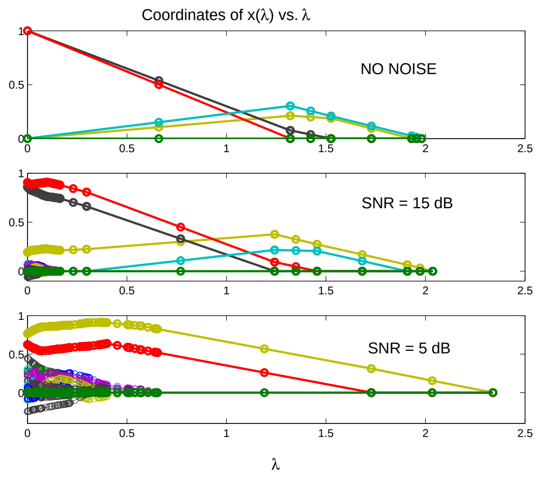

Fig. 2. Solution paths $\hat{\mathbf{x}}(\lambda)$ for all λ with varying levels of noise. $\mathbf{A}$ is 20×100. Top: no noise. Middle: SNR = 15 dB. Bottom: SNR = 5 dB.

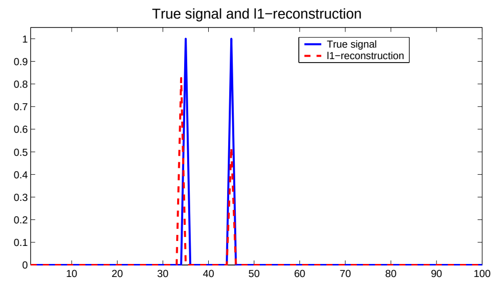

Fig. 3. $\hat{\mathbf{x}}(\lambda)$ for $\lambda = 0.6526$, the minimum λ leading to two non-zero components. SNR = 5 dB.

The above set of experiments were done for a severely overcomplete dictionary ($\mathbf{A}$ is 20×100). Let us now consider a mildly overcomplete, 20×23 DCT dictionary, $\mathbf{A}$. This problem is less demanding than the previous scenario in the sense that the desired signal representation is on a "coarser grid" of dictionary elements (leading to smaller mutual coherence [2]). In Figure 4, we observe that for noisy data the results exhibit excellent stability: even with moderate amounts of noise, SNR = 5 dB, the two non-zero components are clearly visible for any choice of λ. We note that these components exactly match the indices of non-zero elements of $\mathbf{x}$.

Some observations can be drawn from the above experiments. The components of $\hat{\mathbf{x}}(\lambda)$ tend to decrease as λ increases, but as can be seen from the middle plot in Figure 2, a component which was equal to 0 may become non-zero as λ increases. We also observe that sparse representation is easier in dictionaries with well-separated elements (in the sense of [2]). However, all hope is not lost even for severely overcomplete dictionaries, as long as they have certain structure.

5. CONCLUSION

We have described a simple and efficient algorithm to generate entire solution paths (as a function of the regularization parameter) of basis pursuit for sparse signal representation in overcomplete dictionaries. The algorithm can also be used to identify good choices for the regularization parameter. The ease in generating the solution paths make them a useful tool for empirical exploration of the behavior of basis pursuit in various scenarios.

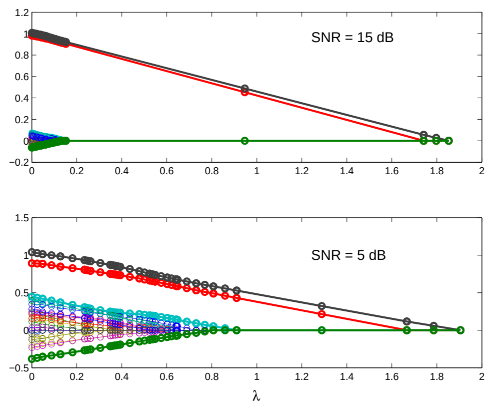

Fig. 4. Solution paths $\hat{\mathbf{x}}(\lambda)$ for all λ with varying levels of noise. $\mathbf{A}$ is 20×23. Top: SNR = 15 dB. Bottom: SNR = 5 dB.

6. REFERENCES

[1] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Review*, vol. 43, no. 1, pp. 129–159, 2001.

[2] D. L. Donoho and M. Elad, "Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ^1 minimization," *Proc. Nat. Aca. Sci.*, vol. 100, pp. 2197–2202, Mar. 2003.

[3] R. Gribonval and M. Nielsen, "Sparse representations in unions of bases," *IEEE Trans. Information Theory*, vol. 49, no. 12, pp. 3320–3325, Dec. 2003.

[4] D. M. Malioutov, M. Çetin, and A. S. Willsky, "Optimal sparse representations in general overcomplete bases," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, May 2004, vol. 2, pp. 793–796.

[5] J.-J. Fuchs, "On sparse representations in arbitrary redundant bases," *IEEE Trans. Information Theory*, vol. 50, no. 6, pp. 1341–1344, June 2004.

[6] J. A. Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Trans. Information Theory*, vol. 50, no. 10, pp. 2231–2242, Oct. 2004.

[7] M. R. Osborne, B. Presnell, and B. A. Turlach, "A new approach to variable selection in least squares problems," *IMA Journal of Numerical Analysis*, vol. 20, no. 3, pp. 389–403, 2000.

[8] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *Annals of Statistics*, vol. 32, no. 2, pp. 407–499, Apr. 2004.

[9] D. M. Malioutov, M. Çetin, and A. S. Willsky, "A sparse signal reconstruction perspective for source localization with sensor arrays," *IEEE Trans. Signal Processing*, to appear.