Sparse Solutions of Systems of Equations and Sparse Modelling of Signals and Images

Alfredo Nava-Tudela
ant@umd.edu
John J. Benedetto
Department of Mathematics
jjb@umd.edu

Abstract

In this project we are interested in finding sparse solutions to systems of linear equations $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{A}$ is underdetermined and fully-ranked, as presented in [1]. In particular, we are interested in the application of this problem to signal compression. We examine the implementation of the *orthogonal matching pursuit* algorithm and explore its application to the above problem.

1 Introduction and context

Let n and m be two positive natural numbers such that $n < m$, and consider a full rank matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$. Given a column vector $\mathbf{b} \in \mathbb{R}^n$, we know that there is an infinite number of solutions to the system of linear equations

$$\mathbf{Ax} = \mathbf{b}, \tag{1}$$

where $\mathbf{x}$ is a column vector in $\mathbb{R}^m$ [5]. That is, there is an infinite number of column vectors $\mathbf{x} \in \mathbb{R}^m$ that satisfy equation (1). However, of this infinite number of possible solutions, we are interested in those solutions that are the *sparsest*. By this we mean solutions where $\mathbf{x}$ has the fewest number of non-zero entries.

Moreover, the relevance of finding sparse solutions to the underdetermined system of linear equations (1) is greater given that there are several well-known signal and image processing problems that can be cast as demanding solutions of underdetermined systems of linear equations [1]. For example, the media encoding standard JPEG [2, 7] and its successor, JPEG-2000 [6], are both based on the notion of transform encoding. The JPEG standard uses the Discrete Cosine Transform (DCT), and the JPEG-2000 standard uses the Discrete Wavelet Transform (DWT). Both JPEG standards use properties of the DCT or the DWT, respectively, to achieve compression by creating approximations that represent the original image in a sparse way. Other important examples of signal and image processing problems where transform sparsity is a driving factor are denoising and image deblurring.

1.1 Defining the problem of finding sparse solutions of $\mathbf{Ax} = \mathbf{b}$

Consider a full-rank matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$ with $n < m$, and define the underdetermined system of linear equations $\mathbf{Ax} = \mathbf{b}$. From the infinite number of solutions, we shall narrow down the choice to a well-defined solution by introducing a real valued function $\mathcal{J}(\mathbf{x})$ to evaluate the desirability of a would-be solution $\mathbf{x} \in \mathbb{R}^m$, with smaller values of $\mathcal{J}$ being preferred. This way, we can define the general optimization problem $(P_{\mathcal{J}})$ as

$$(P_{\mathcal{J}}): \qquad \min_{\mathbf{x}} \mathcal{J}(\mathbf{x}) \quad \text{subject to} \quad \mathbf{Ax} = \mathbf{b}. \tag{2}$$

Selecting a strictly convex function $\mathcal{J}(\cdot)$ guarantees a unique solution. For example if $\mathcal{J}(\mathbf{x}) = ||\mathbf{x}||_2^2$, the squared Euclidean norm of $\mathbf{x}$, the problem (P_2) that results from this choice has the unique minimum-norm solution $\hat{\mathbf{x}}$ given by

$$\hat{\mathbf{x}} = \mathbf{A}^+\mathbf{b} = \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{b}.$$

We know that the squared ℓ^2 norm is a measure of energy; we are interested in measures of *sparsity*. As was mentioned before, a vector $\mathbf{x}$ is sparse if there are few nonzero elements in the possible entries in $\mathbf{x}$. As such we shall introduce the ℓ^0 "norm"

$$||\mathbf{x}||_0 = \#\{i : x_i \neq 0\}.$$

Thus, if $||\mathbf{x}||_0 \ll n$, then $\mathbf{x}$ is sparse.

Consider the problem (P_0) obtained from the general prescription (2) that results from choosing $\mathcal{J}(\mathbf{x}) = ||\mathbf{x}||_0$, viz.,

$$(P_0): \qquad \min_{\mathbf{x}} ||\mathbf{x}||_0 \quad \text{subject to} \quad \mathbf{Ax} = \mathbf{b}. \tag{3}$$

Unfortunately, the discrete and discontinuous nature of the ℓ^0 norm impedes the application of the standard convex analysis ideas that were at the core of the solution of (P_2). Moreover, it has been proven that (P_0) is, in general, NP-hard [3]. However, the solution of (P_0) can still be obtained by *greedy algorithms* when a sufficiently sparse solution exists. We introduce one such greedy algorithm next.

2 Approach

2.1 Orthogonal Matching Pursuit

In the first half of this project, we are interested in implementing one of the many greedy algorithms (GAs) that attempt to solve (P_0). The general idea is as follows. Starting from $\mathbf{x}^0 = \mathbf{0}$, a greedy strategy iteratively constructs a k-term approximation $\mathbf{x}^k$ by maintaining a set of active columns—initially empty—and, at each stage, expanding that set by one additional column. The column chosen at each stage maximally reduces the residual ℓ^2 error in approximating $\mathbf{b}$ from the current set of active columns. After constructing an approximation including the new column, the residual error ℓ^2 is evaluated; if it now falls below a specified threshold, the algorithm terminates.

Orthogonal Matching Pursuit (OMP)—a GA for approximating the solution of (P_0):

Task: Approximate the solution of $(P_0) : \min_{\mathbf{x}} ||\mathbf{x}||_0$ subject to $\mathbf{Ax} = \mathbf{b}$.

Parameters: We are given the matrix $\mathbf{A}$, the vector $\mathbf{b}$, and the threshold ϵ_0.

Initialization: Initialize $k = 0$, and set

- The initial solution $\mathbf{x}^0 = \mathbf{0}$.
- The initial residual $\mathbf{r}^0 = \mathbf{b} - \mathbf{Ax}^0 = \mathbf{b}$.
- The initial solution support $\mathcal{S}^0 = Support\{\mathbf{x}^0\} = \emptyset$.

Main Iteration: Increment k by 1 and perform the following steps:

- **Sweep:** Compute the errors $\epsilon(j) = \min_{z_j} ||z_j\mathbf{a}_j - \mathbf{r}^{k-1}||_2^2$ for all j using the optimal choice $z_j^* = \mathbf{a}_j^T\mathbf{r}^{k-1}/||\mathbf{a}_j||_2^2$.
- **Update Support:** Find a minimizer j_0 of $\epsilon(j)$: $\forall j \notin \mathcal{S}^{k-1},\ \epsilon(j_0) \leq \epsilon(j)$, and update $\mathcal{S}^k = \mathcal{S}^{k-1} \cup \{j_0\}$.
- **Update Provisional Solution:** Compute $\mathbf{x}^k$, the minimizer of $||\mathbf{Ax} - \mathbf{b}||_2^2$ subject to $Support\{\mathbf{x}\} = \mathcal{S}^k$.
- **Update Residual:** Compute $\mathbf{r}^k = \mathbf{b} - \mathbf{Ax}^k$.
- **Stopping Rule:** If $||\mathbf{r}^k||_2 < \epsilon_0$, stop. Otherwise, apply another iteration.

Output: The proposed solution is $\mathbf{x}^k$ obtained after k iterations.

This algorithm is known in the literature of signal processing by the name *orthogonal matching pursuit* (OMP), and this is the algorithm we shall implement, validate, and test.

2.2 Application to signal compression

For the second half of this project, we shall use the OMP algorithm implemented in the first half to study the role of sparse signal analysis in the problem of signal compression.

Suppose we have a signal $\mathbf{y} \in \mathbb{R}^n$ that usually requires a description by n numbers. However, suppose that we can solve

$$(P_0^\epsilon) : \quad \min_{\mathbf{x}} ||\mathbf{x}||_0 \ \text{ subject to } \ ||\mathbf{y} - \mathbf{Ax}||_2 \leq \epsilon, \tag{4}$$

and the solution $\mathbf{x}_0^\epsilon$ has k non-zeros, with $k \ll n$, then we would have obtained an approximation $\hat{\mathbf{y}} = \mathbf{Ax}_\mathbf{0}^\epsilon$ to $\mathbf{y}$ using k scalars, with an approximation error at most ϵ. By increasing ϵ we obtain stronger compression with larger approximation error, and in this way we can obtain a rate-distortion curve for a compression mechanism.

In our case, we shall study matrices of the type $\mathbf{A} = [\mathbf{DCT}\ \mathbf{DWT}]$, where **DCT** and **DWT** represent matrices that implement the Discrete Cosine Transform and the Discrete Wavelet Transform, respectively.

3 Implementation

To implement (OMP), we shall work on an Apple *MacBook Pro* laptop running Mac OS X v10.5, with an *Intel Core 2 Duo* processor running at 2.16 GHz, and 2 GB of memory. The software and language that we shall use to implement the algorithm is *Matlab*.

For a given matrix $\mathbf{A} \in \mathbb{R}^{n\times m}$, if the approximation delivered by (OMP) has k_0 zeros, the method requires $\mathcal{O}(k_0 mn)$ flops in general; this can be dramatically better than the exhaustive search, which requires $\mathcal{O}(nm^{k_0}k_0^2)$ flops.

We would like to make the following observations about the (OMP) algorithm described in section (2.1). The step that updates the provisional solution seeks to minimize $||\mathbf{Ax}-\mathbf{b}||_2^2$, subject to $Support\{\mathbf{x}\} = \mathcal{S}^k$. This is equivalent to solving the least squares approximation problem $\min_{\tilde{\mathbf{x}}} ||\mathbf{A}^{(k)}\tilde{\mathbf{x}} - \mathbf{b}||_2^2$ for the matrix $\mathbf{A}^{(k)}$ that results from using only the k active columns of $\mathbf{A}$ defined by $\mathcal{S}^k$, and $\tilde{\mathbf{x}}$ is the vector in $\mathbb{R}^k$ whose i-th entry corresponds to the column of $\mathbf{A}$ that was chosen in the i-th step of the main iteration. For the case when $\mathbf{A}$ is a relatively small matrix, we can solve this problem, for example, by factorizing $\mathbf{A}^{(k)} = \mathbf{Q}^{(k)}\mathbf{R}^{(k)}$ with the QR-algorithm, and then noting that $\tilde{\mathbf{x}}_0 = \mathbf{R}^k(1:k,:)^{-1}\mathbf{c}^{(k)}$, where $\mathbf{c}^{(k)} = (\mathbf{Q}^{(k)T}\mathbf{b})(1:k)$, is the solution to the equivalent minimization problem described above.

However, if $\mathbf{A}$ is too big to store in memory the QR decompositions described above, we could, for instance, recourse to iterative methods, such as the conjugate gradient method, where only matrix-vector multiplications would be required, and for which the implicit storage of the matrix is not necessary.

4 Databases

As mentioned in the approach section (2) above, the project will be divided in two parts. The first part deals with the implementation and validation of (OMP), and the second part deals with its testing, where we will seek to reproduce the results that pertain to (OMP) in section (3.3.1) of [1], on the one hand; and a test suite in signal compression for which we use (OMP), on the other hand.

For the first half of the project, the database to validate the algorithm implementation will be a synthetic set of "right hand side" vectors $\mathbf{b}$ that will come from known sparse vectors $\mathbf{x}$ that are sparse enough as to guarantee the theoretical convergence of the (OMP) algorithm. For the second half, we will follow the setup in section (3.3.1) of [1] for the (OMP) algorithm part of the experiment; and for the signal compression test suite, we will select a diverse set of real life signals (which could be either real life images, or sound) arranged in a 1-dimensional vector format apt for processing.

5 Validation

Given a matrix $\mathbf{A} \in \mathbb{R}^{n\times m}$ with $n < m$, we can compute its mutual coherence defined as follows,

Definition 1. *The mutual coherence of a given matrix* $\mathbf{A}$ *is the largest absolute normalized inner product between different columns from* $\mathbf{A}$*. Denoting the k-th column in* $\mathbf{A}$ *by* $\mathbf{a}_k$*, the mutual coherence is given by*

$$\mu(\mathbf{A}) = \max_{1\leq k,j\leq m,\ k\neq j} \frac{|\mathbf{a}_k^T\mathbf{a}_j|}{||\mathbf{a}_k||_2 \cdot ||\mathbf{a}_j||_2}. \tag{5}$$

It can be proven that (OMP) converges to the sparsest solution $\mathbf{x}^*$ when that solution satisfies $||\mathbf{x}^*||_0 < 1+1/\mu(\mathbf{A})$. To test our implementation we can generate a series of random matrices $\mathbf{A}$ of size 100×200, with entries independently drawn at random from a Gaussian distribution of zero mean and unit variance, $\mathcal{N}(0,1)$, and compute their mutual coherence $\mu(\mathbf{A})$. This way, we can compute how sparse to make test vectors $\mathbf{x}$ to generate right hand side vectors $\mathbf{b}$ to feed to our algorithm. Then, if given a tolerance ϵ_0, the algorithm consistently returns a sparse solution with the same number of nonzero entries as the test vector, and the solution returned is within ϵ_0 of the test vector, we would have validated our implementation.

6 Testing

For the first portion of our testing protocol, we will reproduce the experiment described in section (3.3.1) of [1], limited to the results obtained using (OMP) alone.

Consider a random matrix $\mathbf{A}$ of size 100×200, with entries independently drawn at random from a Gaussian distribution of zero mean and unit variance, $\mathcal{N}(0,1)$. It can be proven that, with probability 1, every solution for the system $\mathbf{Ax} = \mathbf{b}$ with less than 51 entries is necessarily the sparsest one possible, and, as such, it is the solution of (P_0). By randomly generating such sufficiently sparse vectors $\mathbf{x}$ (choosing the nonzero locations uniformly over the support in random and their values from $\mathcal{N}(0,1)$), we generate vectors $\mathbf{b}$. This way, we know the sparsest solution to $\mathbf{Ax} = \mathbf{b}$, and we shall be able to compare this to the results given by (OMP).

Since we are trying to reproduce the results that pertain to (OMP) in Figure 2 of page 56 of [1], we will consider cardinalities in the range of 1 to 70—even though we know that, with probability 1, only those solutions with cardinality equal to or less than 51 are uniquely the sparsest ones possible—, and we will conduct 100 repetitions and average the results as to obtain the probability of the algorithm finding the solution with which we generated the right hand side $\mathbf{b}$. We will compare our results with results obtained by published (OMP) implementations, e.g., [4].

For our second test suite, we will study the compression properties of matrices $\mathbf{A}$ that concatenate the Discrete Fourier Transform (DCT) and the Discrete Wavelet Transform (DWT), and compare to the compression properties that the DCT or DWT achieve alone. We shall proceed in the following way. Suppose that we have a real life test signal $\mathbf{s}$, whether it be an image or a sound bite, and compute its DCT and DWT transforms, say $\tilde{\mathbf{s}}_{DCT}$ and $\tilde{\mathbf{s}}_{DWT}$, respectively. Then choose the k largest coefficients in absolute value for each transform, and rebuild the signal from those coefficients alone. Let $\mathbf{s}_{DCT}(k)$ and $\mathbf{s}_{DWT}(k)$ be those reconstructions. Next, compute the ℓ_2 approximation error between each reconstruction and the original, say $e_{DCT}(k) = ||\mathbf{s} - \mathbf{s}_{DCT}(k)||_2$ and $e_{DWT}(k) = ||\mathbf{s} - \mathbf{s}_{DWT}(k)||_2$, respectively. Finally, we run twice (OMP) using as inputs our matrix $\mathbf{A}$, our right hand side vector $\mathbf{b} = \mathbf{s}$, and, for the first run, $\epsilon_0 = e_{DCT}(k)$ and, for the second run, $\epsilon_0 = e_{DWT}(k)$, corresponding to the errors that resulted from reconstructing $\mathbf{s}$ using the k largest coefficients in absolute value for the DCT and the DWT, respectively.

Let $\mathbf{s}_{OMP}(e_{DCT}(k))$ and $\mathbf{s}_{OMP}(e_{DWT}(k))$ be the solutions returned by our two runs of (OMP). Ideally, we would like to have both $||\mathbf{s}_{OMP}(e_{DCT}(k))||_0$ and $||\mathbf{s}_{OMP}(e_{DWT}(k))||_0$ be

less than k. We then graph k-vs-error for comparison purposes.

7 Project schedule

We propose to guide our work with the following schedule, with the understanding that we may take on new tasks if we find ourselves ahead of schedule, of course.

- **Oct. 18, 2010:** Complete project proposal and submit for final review.
- **Oct. 18 - Nov. 5:** Make revisions to the project proposal, if necessary, and implement (OMP).
- **Nov. 8 - Nov. 26:** Validate (OMP).
- **Nov. 29 - Dec. 3:** Write mid-year report.
- **Dec. 6 - Dec. 10:** Prepare mid-year oral presentation.
- Some time after that, give mid-year oral presentation.
- **Jan. 24 - Feb. 11:** Testing of (OMP). Reproduce paper results.
- **Jan. 14 - Apr. 8:** Testing of (OMP) on $\mathbf{A} = [\mathbf{DCT}\ \mathbf{DWT}]$.
- **Apr. 11 - Apr. 22:** Write final report.
- **Apr. 25 - Apr. 29:** Prepare final oral presentation.
- Some time after that, give final oral presentation

8 Milestones

We foresee the following milestones in our project, with the dates proposed built around what needs to happen and the scheduled we have proposed above.

- **October 29:** (OMP) implemented for small matrices $\mathbf{A}$ for which a QR decomposition can be stored and used to solve the minimization described in the main iteration where we update the provisional solution.
- **November 5:** (OMP) implemented for both small and large $\mathbf{A}$, using the conjugate gradient method in the case of large $\mathbf{A}$, and ready to be validated.
- **November 17:** Validation on small matrices completed.
- **November 26:** Validation on large matrices completed.
- **December 3:** Finish writing the mid-year report.
- **February 4:** Have partial results on tests that replicate (3.3.1) of [1].

- **February 11:** Complete tests that replicate (3.3.1) of [1].
- **February 12 – April 8:** We have to complete the following 4 stages at one time or another during this period:
  - ◇ Select the wavelets to use for the DWT in the signal compression test suite. Select the images or signals to compress.
  - ◇ Form or compute, as needed and justified, the respective **DCT**, **DWT**, and $\mathbf{A} = [\mathbf{DCT}\ \mathbf{DWT}]$ matrices to conduct the second suite of experiments described in section (6).
  - ◇ Obtain graphs of k-vs-error for the different schemes tested.
- April 22: Final report written.

9 Deliverables

Upon completion of this project, on or before the date stipulated to submit final results[1], we shall provide (1) all code in *Matlab* used to produce all the results, (2) a graph reproducing the results of section (3.3.1) in [1] that pertain to (OMP), (3) graphs of compression-vs-error for the signal processing application portion of this project, and (4) any other graph that we may deem of interest that would have been produced during the development, validation, and testing of (OMP). We shall also submit (5) a mid-year report, and (6) a final report, as described in our schedule.

References

[1] A. M. Bruckstein, D. L. Donoho, and M. Elad, *From sparse solutions of systems of equations to sparse modeling of signals and images*, SIAM Review, 51 (2009), pp. 34–81.

[2] S. Mallat, *A Wavelet Tour of Signal Processing*, Academic Press, 1998.

[3] B. K. Natarajan, *Sparse approximate solutions to linear systems*, SIAM Journal on Computing, 24 (1995), pp. 227–234.

[4] SparseLab. http://sparselab.stanford.edu/.

[5] G. W. Stewart, *Introduction to Matrix Computations*, Academic Press, 1973.

[6] D. S. Taubman and M. W. Mercellin, *JPEG 2000: Image Compression Fundamentals, Standards and Practice*, Kluwer Academic Publishers, 2001.

[7] G. K. Wallace, *The JPEG still picture compression standard*, Communications of the ACM, 34 (1991), pp. 30–44.

[1] This date is yet to be decided as of October 18, 2010, but we shall assume it will be consistent with the project schedule proposed herein.