# Introduction to Sparsity in Signal Processing[1]

Ivan Selesnick June 21, 2012 NYU-Poly

## 1 Introduction

These notes describe how sparsity can be used in several signal processing problems. A common theme throughout these notes is the comparison between the least square solution and the sparsity-based solution. In each case, the sparsity-based solution has a clear advantage. It must be emphasized that this will not be true in general, unless the signal to be processed is either sparse or has a sparse representation with respect to a known transform.

To keep the explanations as clear as possible, the examples given in these notes are restricted to 1-D signals. However, the concepts, problem formulations, and algorithms can all be used in the multidimensional case as well.

For a broader overview of sparsity and compressed sensing, see the June 2010 special issue of the *Proceedings of the IEEE* [5], the March 2006 issue of *Signal Processing* [15], or the 2011 workshop SPARS11 [22].

### 1.1 Underdetermined equations

Consider a system of under-determined system of equations

$$\mathbf{y} = \mathbf{A}\mathbf{x} \tag{1}$$

where $\mathbf{A}$ is an $M \times N$ matrix, $\mathbf{y}$ is a length-$M$ vector, and $\mathbf{x}$ is a length-$N$ vector, where $N > M$.

$$\mathbf{y} = \begin{bmatrix} y(0) \\ \vdots \\ y(M-1) \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} x(0) \\ \vdots \\ x(N-1) \end{bmatrix}$$

[1] For feedback/corrections, email selesi@poly.edu.

Support from NSF under Grant CCF-1018020 is gratefully acknowledged.

Matlab software to reproduce the examples in these notes is available on the web or from the author. http://eeweb.poly.edu/iselesni/teaching/lecture_notes/sparsity_intro/index.html

The system has more unknowns than equations. The matrix $\mathbf{A}$ is wider than it is tall. We will assume that $\mathbf{A}\mathbf{A}^*$ is invertible, therefore the system of equations (1) has infinitely many solutions.

We will use the $\ell_2$ and $\ell_1$ norms, which are defined by:

$$\|\mathbf{x}\|_2^2 := \sum_{n=0}^{N-1} |x(n)|^2 \tag{2}$$

$$\|\mathbf{x}\|_1 := \sum_{n=0}^{N-1} |x(n)|. \tag{3}$$

Often, $\|\mathbf{x}\|_2^2$, i. e. the sum of squares, is referred to as the 'energy' of $\mathbf{x}$.

### 1.2 Least squares

A common method to solve (1) is to minimize the energy of $\mathbf{x}$.

$$\underset{\mathbf{x}}{\operatorname{argmin}} \ \|\mathbf{x}\|_2^2 \tag{4a}$$

$$\text{such that } \mathbf{y} = \mathbf{A}\mathbf{x}. \tag{4b}$$

The solution to (4) is given explicitly as:

$$\mathbf{x} = \mathbf{A}^*(\mathbf{A}\mathbf{A}^*)^{-1}\,\mathbf{y}. \tag{5}$$

When $\mathbf{y}$ is noisy, then it is not desired that the equations (1) be solved exactly. In this case, a common method to approximately solve (1) is to minimize the cost function:

$$\underset{\mathbf{x}}{\operatorname{argmin}} \ \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_2^2. \tag{6}$$

The solution to (6) is given explicitly by

$$\mathbf{x} = (\mathbf{A}^*\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{A}^*\,\mathbf{y}. \tag{7}$$

Note that each of (5) and (7) calls for the solution to a system of linear equations. In signal processing, the equations may be very large due to $\mathbf{y}$ and/or $\mathbf{x}$ being long signals (or images, or higher dimensional data). For practical algorithms, it is usually necessary to have fast efficient methods to solve these systems of equations. In many cases, the system of equations have special properties or structure that can be exploited for fast solution. For example, sometimes the Fourier transform can be used. Otherwise, iterative methods such as the conjugate gradient algorithm can be used.

### 1.3 Sparse solutions

Another approach to solve (1) is to minimize the sum of absolute values of $\mathbf{x}$. Namely, to solve the optimization problem:

$$\underset{\mathbf{x}}{\operatorname{argmin}} \|\mathbf{x}\|_1 \tag{8a}$$

$$\text{such that } \mathbf{y} = \mathbf{A}\mathbf{x} \tag{8b}$$

where $\|\mathbf{x}\|_1$ is the $\ell_1$ norm of $\mathbf{x}$, defined in (3). Problem (8) is known as the *basis pursuit* (BP) problem [11]. Unlike the least-square problem (4), the solution to the BP problem (8) can not be written in explicit form. The solution can be found only by running an iterative numerical algorithm.

When $\mathbf{y}$ is noisy, then it does not make sense to solve (1) exactly, as noted above. In this case, an approximate solution can be found by minimizing the cost function

$$\underset{\mathbf{x}}{\operatorname{argmin}} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_1. \tag{9}$$

Problem (9) is known as the *basis pursuit denoising* (BPD) problem. Like (8), the BPD problem can not be solved in explicit form. It can be solved only using an iterative numerical algorithm. The BPD problem is also referred to as the 'lasso' problem (least absolute shrinkage and selection operator) [25]. The lasso problem is equivalent to (9), except that it is defined via a constraint on the $\ell_1$ norm.

Even though (4) and (8) are quite similar, it turns out that the solutions are quite different. Likewise, even though (6) and (9) are quite similar, the solutions are quite different. In (4) the values $|x(n)|^2$ are penalized, while in (8) the values $|x(n)|$ are penalized. The only difference is the squaring operation. Note that, when a set of values are squared and then summed to get $s$, the sum $s$ is most sensitive to the largest values, as illustrated in Fig. 1. Therefore, when minimizing $\|\mathbf{x}\|_2^2$ it is especially important that the largest values of $\mathbf{x}$ be made small as they count much more than the smallest values. For this reason, solutions obtained by minimizing $\|\mathbf{x}\|_2^2$ usually have many small values, as they are relatively unimportant. Consequently, the least square solution is usually not *sparse*.

Therefore, when it is known or expected that the solution $\mathbf{x}$ is sparse, it is advantageous to use the $\ell_1$ norm instead of the $\ell_2$ norm. Some examples will be shown in the following sections.

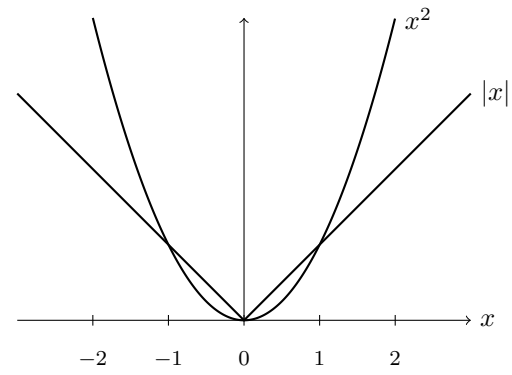


Figure 1: The functions $|x|$ and $x^2$.

### 1.4 Algorithms for sparse solutions

The solution to the basis pursuit problem (8) and basis pursuit denoising problem (9) can not be found using any explicit formula. These problems can be solved only by running iterative numerical algorithms. Note that $\|\mathbf{x}\|_1$ is not differentiable because it is defined as the sum of absolute values. The absolution value function is not differentiable at zero. The fact that the cost functions (8a) and (9) are non-differentiable makes solving the BP and BPD problems somewhat difficult, especially for large-scale problems. However, the BP and BPD problems are *convex* optimization problems, and due to this property, there will not be extraneous local minima. In addition, there are many theoretical results and practical algorithms from optimization theory developed specifically for convex optimization problems [9].

Several different algorithms have been developed for solving the BP and BPD problems [8,12]. There has been some effort by different groups to develop good algorithms for these and related algorithms. A good algorithm is one that converges fast to the solution and has low computational cost. The speed of convergence can be measured in terms of the number of iterations or in terms of execution time. An early algorithm to solve the BPD problem is the 'iterative shrinkage/thresholding algorithm' (ISTA) [12–14],

which has the property that the cost function is guaranteed to decrease on each iteration of the algorithm. However, ISTA converges slowly for some problems. A more recent algorithm is 'fast ISTA' (FISTA) [6]; although the cost function may increase on some iterations, FISTA has much faster convergence than ISTA. FISTA has the unusually good property of having quadratic convergence rate. Another recent algorithm is the 'split variable augmented Lagrangian shrinkage algorithm' (SALSA) [1,2]. SALSA often has very good convergence properties in practice. SALSA requires solving a least square problem at each iteration, so it is not always practical. However, in many cases (as in the examples below) the least square problem can be solved very easily, and in these cases SALSA is a very effective algorithm.

For signal processing applications, it is also desired of algorithms for BP and BPD that they do not require the matrix $\mathbf{A}$ to be explicitly stored, but instead require only functions to multiply $\mathbf{A}$ and $\mathbf{A}^*$ with vectors. An algorithm of this kind is sometimes called 'matrix-free'. For example, if $\mathbf{A}$ is a filter convolution matrix, then the matrix-vector product $\mathbf{Ax}$ can be computed using fast FFT-based convolution algorithms or using a difference equation. That is faster than performing full matrix-vector multiplication. It also saves memory: it is necessary to store only the impulse response or frequency response of the filter — the actual matrix $\mathbf{A}$ itself is never needed. This is especially important in signal processing because the matrix $\mathbf{A}$ would often be extremely large. For a large matrix $\mathbf{A}$, $(i)$ a lot of storage memory is needed and $(ii)$ computing the matrix-vector product $\mathbf{Ax}$ is slow. Hence matrix-free algorithms are especially important when $\mathbf{A}$ is large.

**Parseval frames**

If the matrix $\mathbf{A}$ satisfies the equation

$$\mathbf{A}\mathbf{A}^* = p\mathbf{I} \tag{10}$$

for some positive number $p$, then the columns of $\mathbf{A}$ are said to form a 'Parseval frame' (or 'tight frame'). This kind of matrix is like a unitary (or orthonormal) matrix, except here $\mathbf{A}$ can be rectangular. (A unitary matrix must be square.)

Note that a rectangular matrix can not have an inverse, but it can have either a left or right inverse. If a rectangular matrix $\mathbf{A}$ satisfies (10), then the right inverse of $\mathbf{A}$ is $\mathbf{A}^*$ ($\mathbf{A}$ has no left inverse).

As a special case of (5), if $\mathbf{A}$ satisfies the equation (10) then (5) becomes

$$\mathbf{x} = \mathbf{A}^*(\mathbf{A}\mathbf{A}^*)^{-1}\,\mathbf{y} = \frac{1}{p}\mathbf{A}^*\,\mathbf{y} \qquad (\mathbf{A}\mathbf{A}^* = p\mathbf{I}) \tag{11}$$

which is computationally more efficient than (5) because no matrix inversion is needed.

As a special case of (7), if $\mathbf{A}$ satisfies the equation (10), then (7) becomes

$$\mathbf{x} = (\mathbf{A}^*\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{A}^*\,\mathbf{y} = \frac{1}{\lambda + p}\,\mathbf{A}^*\,\mathbf{y} \qquad (\mathbf{A}\mathbf{A}^* = p\mathbf{I}) \tag{12}$$

which is computationally more efficient than (7) because, again, no matrix inversion is needed. To derive (12) from (7) the *'matrix inverse lemma'* can be used.

If $\mathbf{A}$ satisfies (10), then as (11) and (12) show, it is very easy to find least square solutions. It turns out that if $\mathbf{A}$ satisfies (10), then some algorithms for BP and BPD also become computationally easier.

### 1.5 Exercises

1. Prove (5).

2. Prove (7).

3. Prove (12). The 'matrix inverse lemma' can be used.

## 2 Sparse Fourier coefficients using BP

The Fourier transform of a signal tells how to write the signal as a sum of sinusoids. In this way, it tells what frequencies are present in a signal. However, the Fourier transform is not the only way to write a signal as a sum of sinusoids, especially for discrete signals of finite length. In the following example, it will be shown that basis pursuit ($\ell_1$ norm minimization) can be used to obtain a frequency spectrum more sparse than that obtained using the Fourier transform.

Suppose that the $M$-point signal $y(m)$ is written as

$$y(m) = \sum_{n=0}^{N-1} c(n) \exp\left(\mathrm{j}\frac{2\pi}{N}mn\right), \qquad 0 \le m \le M-1 \tag{13}$$

where $c(n)$ is a length-$N$ coefficient sequence, with $M \le N$. In vector notation, the equation (13) can be expressed as

$$\mathbf{y} = \mathbf{A}\mathbf{c} \tag{14}$$

where $\mathbf{A}$ is a matrix of size $M \times N$ given by

$$\mathbf{A}_{m,n} = \exp\left(\mathrm{j}\frac{2\pi}{N}mn\right), \quad 0 \le m \le M-1, \;\; 0 \le n \le N-1 \tag{15}$$

and $\mathbf{c}$ is a length-$N$ vector. The coefficients $c(n)$ are frequency-domain (Fourier) coefficients.

Some remarks regarding the matrix $\mathbf{A}$ defined in (15):

1. If $N = M$, then $\mathbf{A}$ is the inverse $N$-point DFT matrix (except for a normalization constant).

2. If $N > M$, then $\mathbf{A}$ is the first $M$ rows of the inverse $N$-point DFT matrix (except for a normalization constant). Therefore, multiplying a vector by $\mathbf{A}$ or $\mathbf{A}^*$ can be done very efficiently using the FFT. For example, in Matlab, $\mathbf{y} = \mathbf{A}\mathbf{c}$ can be implemented using the function:

```
function y = A(c, M, N)
    v = N * ifft(c);
    y = v(1:M);
end
```

   Similarly, $\mathbf{A}^*\mathbf{y}$ can be obtained by zero-padding and computing the DFT. In Matlab, we can implement $\mathbf{c} = \mathbf{A}^*\mathbf{y}$ as the function:

```
function c = AT(y, M, N)
    c = fft([y; zeros(N-M, 1)]);
end
```

   Because the matrices $\mathbf{A}$ and $\mathbf{A}^*$ can be multiplied with vectors efficiently using the FFT without actually creating or storing the matrix $\mathbf{A}$, matrix-free algorithms can be readily used.

3. Due to the orthogonality properties of complex sinusoids, the matrix $\mathbf{A}$ satisfies:

$$\mathbf{A}\mathbf{A}^* = N\,\mathbf{I}_M \tag{16}$$

   so

$$(\mathbf{A}\mathbf{A}^*)^{-1} = \frac{1}{N}\mathbf{I}_M. \tag{17}$$

   That is, the matrix $\mathbf{A}$ defined in (15) satisfies (10).

When $N = M$, then the coefficients $\mathbf{c}$ satisfying (14) are uniquely determined and can be found using the discrete Fourier transform (DFT). But when $N > M$, there are more coefficients than signal values and the coefficients $\mathbf{c}$ are not unique. Any vector $\mathbf{c}$ satisfying $\mathbf{y} = \mathbf{A}\mathbf{c}$ can be considered a valid set of coefficients. To find a particular solution we can minimize either $\|\mathbf{c}\|_2^2$ or $\|\mathbf{c}\|_1$. That is, we can solve either the least square problem:

$$\underset{\mathbf{c}}{\operatorname{argmin}} \; \|\mathbf{c}\|_2^2 \tag{18a}$$

$$\text{such that } \mathbf{y} = \mathbf{A}\mathbf{c} \tag{18b}$$

or the basis pursuit problem:

$$\underset{\mathbf{c}}{\operatorname{argmin}} \; \|\mathbf{c}\|_1 \tag{19a}$$

$$\text{such that } \mathbf{y} = \mathbf{A}\mathbf{c}. \tag{19b}$$

The two solutions can be quite different, as illustrated below.

From Section 1.2, the solution to the least square problem (18) is given by

$$\mathbf{c} = \mathbf{A}^*(\mathbf{A}\mathbf{A}^*)^{-1}\mathbf{y} = \frac{1}{N}\mathbf{A}^*\mathbf{y} \qquad \text{(least square solution)} \tag{20}$$

where we have used (17). As noted above, $\mathbf{A}^*\mathbf{y}$ is equivalent to zero-padding the length-$M$ signal $\mathbf{y}$ to length-$N$ and computing its DFT. Hence, the least square solution is simply the DFT of the zero-padded signal. The solution to the basis pursuit problem (19) must be found using an iterative numerical algorithm.

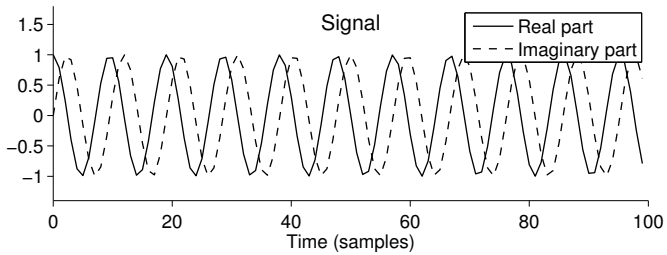


Figure 2: The signal is a complex sinusoid.

**Example**

A simple length-100 signal $\mathbf{y}$ is shown in Fig. 2. The signal is given by:

$$y(m) = \exp\left(\mathrm{j}2\pi f_o m/M\right), \quad 0 \le m \le M-1, \quad f_0 = 10.5,\ M = 100.$$

This signal is 10.5 cycles of a complex sinusoid (not a whole number of cycles). If $N = 100$, then the coefficients $\mathbf{c}$ satisfying $\mathbf{y} = \mathbf{A}\mathbf{c}$ are unique (and can be found using the DFT) and the unique $\mathbf{c}$ is illustrated in Fig. 3a. It can be seen that even though the signal $y(m)$ consists of a single frequency, the Fourier coefficients are spread out. This is the famous leakage phenomenon; it occurs whenever a sinusoidal signal has a fractional number of cycles (here, 10.5 cycles).

To obtain a higher resolution frequency spectrum, we can use more Fourier coefficients than signal values; that is, $N > M$. We will use $N = 256$, more than twice $M$ and also a power of two (so a fast radix-2 FFT can be used). In this case, because $N > M$, the coefficients are not unique. The least square solution, easily computed using (20), is shown in Fig. 3b. Note that the least square solution also exhibits the leakage phenomenon.

To obtain the solution to the basis pursuit problem (19), we have used 100 iterations of an iterative algorithm (SALSA). The basis pursuit solution is shown in Fig. 3c. It is clear in the figure that the basis pursuit solution is more sparse than the least square solution. The basis pursuit solution does not have the leakage phenomenon. The cost function history of the SALSA algorithm for this example is illustrated in Fig. 4. It can be seen that the cost function is not strictly decreasing; however, it does flatten out eventually as it reaches the minimum value. It is informative to look at the cost function history of an iterative algorithm in order to evaluate its convergence behavior.

### 2.1 Exercises

1. Show that for the matrix $\mathbf{A}$ defined in (15), the matrix-vector products $\mathbf{A}\mathbf{x}$ and $\mathbf{A}^*\mathbf{y}$ can be computed using the DFT as described.

2. Show that the matrix $\mathbf{A}$ defined in (15) satisfies (16).

3. Reproduce the example (or similar). Also, repeat the example with different frequencies and signal lengths. Comment on your observations.

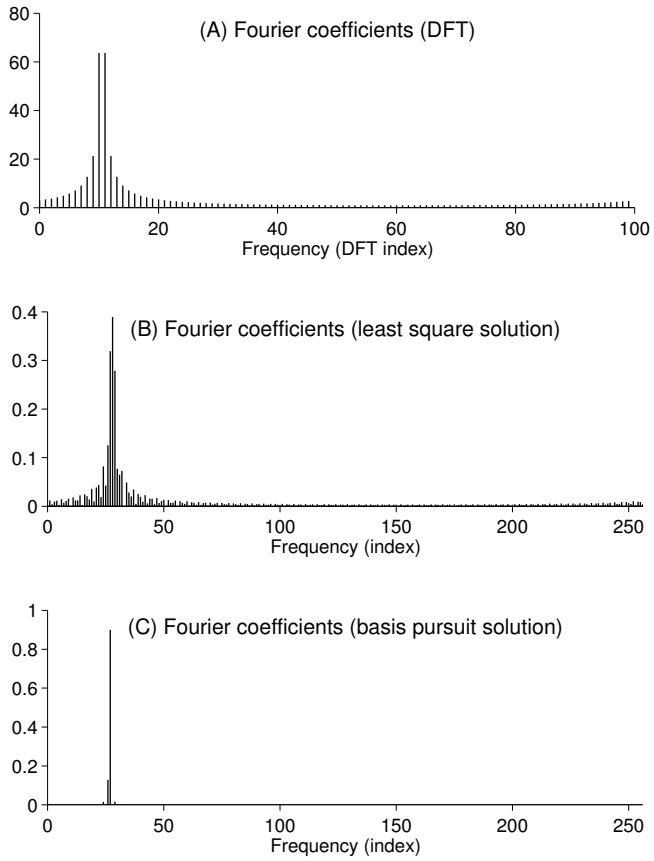


Figure 3: Fourier coefficients of the signal in Fig. 2, calculated three different ways.

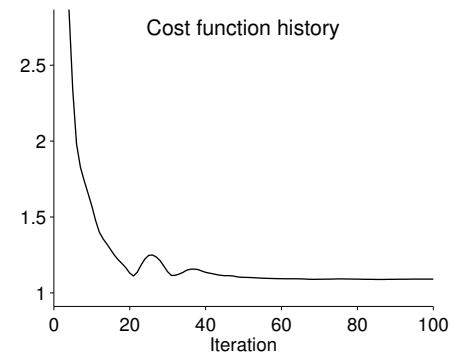


Figure 4: Cost function history of algorithm for basis pursuit solution in Fig. 3c.

## 3 Denoising using BPD

Digital LTI filters are often used for noise reduction (denoising). If the noise and signal occupy separate frequency bands, and if these frequency bands are known, then an appropriate digital LTI filter can be readily designed so as to remove the noise very effectively. But when the noise and signal overlap in the frequency domain, or if the respective frequency bands are unknown, then it is more difficult to do noise filtering using LTI filters. However, if it is known that the signal to be estimated has sparse (or relatively sparse) Fourier coefficients, then sparsity methods can be used as an alternative to LTI filters for noise reduction, as will be illustrated.

A 500 sample noisy speech waveform is illustrated in Fig. 5a. The speech signal was recorded at a sampling rate of 16,000 samples/second, so the duration of the signal is 31 msec. We write the noisy speech signal $y(m)$ as

$$y(m) = s(m) + w(m), \qquad 0 \le m \le M-1, \quad M = 500 \tag{21}$$

where $s(m)$ is the noise-free speech signal and $w(m)$ is the noise sequence. We have used zero-mean Gaussian IID noise to create the noisy speech signal in Fig. 5a. The frequency spectrum of $\mathbf{y}$ is illustrated in Fig. 6a. It was obtained by zero-padding the 500-point signal $\mathbf{y}$ to length 1024 and computing the DFT. Due to the DFT coefficients being symmetric (because $\mathbf{y}$ is real), only the first half of the DFT is shown in Fig. 6. It can be seen that the noise is not isolated to any specific frequency band.

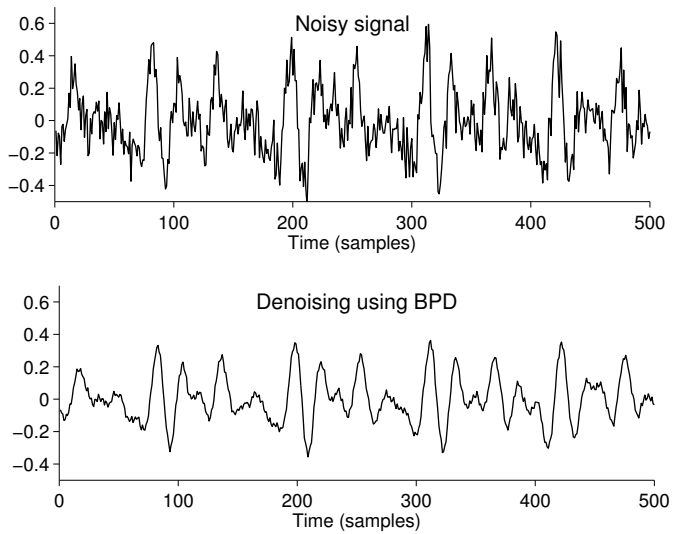


Figure 5: Speech waveform: 500 samples at a sampling rate of 16,000 samples/second.

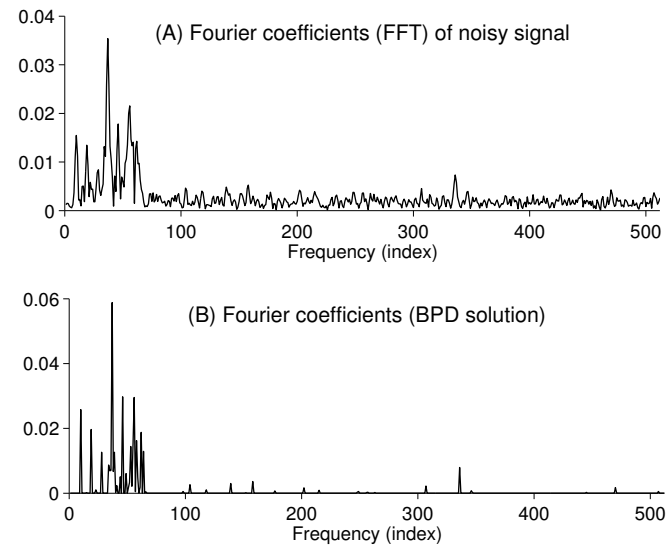


Figure 6: Speech waveform Fourier coefficients. (a) FFT of noisy signal. (b) Fourier coefficients obtained by BPD.

Because the noise is IID, it is uniformly spread in frequency.

Let us assume that the noise-free speech signal $s(n)$ has a sparse set of Fourier coefficients. Then, it is useful to write the noisy speech signal as:

$$\mathbf{y} = \mathbf{A}\mathbf{c} + \mathbf{w}$$

where $\mathbf{A}$ is the $M \times N$ matrix defined in (15). The noisy speech signal is the length-$M$ vector $\mathbf{y}$, the sparse Fourier coefficients is the length-$N$ vector $\mathbf{c}$, and the noise is the length-$M$ noise vector $\mathbf{w}$. Because $\mathbf{y}$ is noisy, it is suitable to find $\mathbf{c}$ by solving either the least square problem

$$\underset{\mathbf{c}}{\operatorname{argmin}} \|\mathbf{y} - \mathbf{A}\mathbf{c}\|_2^2 + \lambda \|\mathbf{c}\|_2^2 \tag{22}$$

or the basis pursuit denoising (BPD) problem

$$\underset{\mathbf{c}}{\operatorname{argmin}} \|\mathbf{y} - \mathbf{A}\mathbf{c}\|_2^2 + \lambda \|\mathbf{c}\|_1. \tag{23}$$

Once $\mathbf{c}$ is found (either the least square or BPD solution), an estimate of the speech signal is given by $\hat{\mathbf{s}} = \mathbf{A}\mathbf{c}$. The parameter $\lambda$ must be set to an appropriate positive real value according to the noise variance. When the noise has a low variance, then $\lambda$ should be chosen to be small. When the noise has a high variance, then $\lambda$ should be chosen to be large.

Consider first the solution to the least square problem (22). From Section 1.2 the least square solution is given by

$$\mathbf{c} = (\mathbf{A}^*\mathbf{A} + \lambda \mathbf{I})^{-1}\mathbf{A}^*\mathbf{y} = \frac{1}{\lambda + N}\mathbf{A}^*\mathbf{y} \qquad (\mathbf{A}\mathbf{A}^* = N\,\mathbf{I}) \tag{24}$$

where we have used (12) and (16). Hence, the least square estimate of the speech signal is given by

$$\hat{\mathbf{s}} = \mathbf{A}\mathbf{c} = \frac{N}{\lambda + N}\mathbf{y} \qquad \text{(least square solution).}$$

Note that as $\lambda$ goes to zero (low noise case), then the least square solution approaches the noisy speech signal. This is appropriate: when there is (almost) no noise, the estimated speech signal should be (almost) the observed signal $\mathbf{y}$. As $\lambda$ increases (high noise case), then the least square solution approaches to zero — the noise is attenuated, but so is the signal.

But the estimate $\hat{\mathbf{s}} = \frac{N}{\lambda+N}\,\mathbf{y}$ is only a scaled version of the noisy signal! No real filtering has been achieved. We do not illustrate the least square solution in Fig. 5, because it is the same as the noisy signal except for scaling by $N/(\lambda + N)$.

For the basis pursuit denoising problem (23), we have used $N = 1024$. Hence, the Fourier coefficient vector $\mathbf{c}$ will be of length 1024. To solve the BPD problem, we used 50 iterations of the algorithm SALSA. The Fourier coefficient vector $\mathbf{c}$ obtained by solving BPD is illustrated in Fig. 6b. (Only the first 512 values of $\mathbf{c}$ are shown because the sequence is symmetric.) The estimated speech signal $\hat{\mathbf{s}} = \mathbf{A}\mathbf{c}$ is illustrated in Fig. 5b. It is clear that the noise has been successfully reduced while the shape of the underlying waveform has been maintained. Moreover, the high frequency component at frequency index 336 visible in Fig. 6 is preserved in the BPD solution, whereas a low-pass filter would have removed it.

Unlike the least square solution, the BPD solution performs denoising as desired. But it should be emphasized that the effectiveness of BPD depends on the coefficients of the signal being *sparse* (or approximately sparse). Otherwise BPD will not be effective.

### 3.1 Exercises

1. Reproduce the speech denoising example in this section (or a similar example). Vary the parameter $\lambda$ and observe how the BPD solution changes as a function of $\lambda$. Comment on your observations.

## 4 Deconvolution using BPD

In some applications, the signal of interest $x(m)$ is not only noisy but is also distorted by an LTI system with impulse response $h(m)$. For example, an out-of-focus image can be modeled as an in-focus image convolved with a blurring function (point-spread function). For convenience, the example below will involve a one-dimensional signal. In this case, the available data $y(m)$ can be written as

$$y(m) = (h * x)(m) + w(m) \tag{25}$$

where '$*$' denotes convolution (linear convolution) and $w(m)$ is additive noise. Given the observed data $y$, we aim to estimate the signal $x$. We will assume that the sequence $h$ is known.

If $x$ is of length $N$ and $h$ is of length $L$, then $y$ will be of length $M = N + L - 1$. In vector notation, (25) can be written as

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{w} \tag{26}$$

where $\mathbf{y}$ and $\mathbf{w}$ are length-$M$ vectors and $\mathbf{x}$ is a length-$N$ vector. The matrix $\mathbf{H}$ is a convolution (Toeplitz) matrix with the form

$$\mathbf{H} = \begin{bmatrix} h_0 & & & \\ h_1 & h_0 & & \\ h_2 & h_1 & h_0 & \\ & h_2 & h_1 & h_0 \\ & & h_2 & h_1 \\ & & & h_2 \end{bmatrix}. \tag{27}$$

The matrix $\mathbf{H}$ is of size $M \times N$ with $M > N$ (because $M = N + L - 1$).

Because of the additive noise $\mathbf{w}$, it is suitable to find $\mathbf{x}$ by solving either the least square problem:

$$\underset{\mathbf{x}}{\operatorname{argmin}}\ \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_2^2 \tag{28}$$

or the basis pursuit denoising problem:

$$\underset{\mathbf{x}}{\operatorname{argmin}}\ \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_1. \tag{29}$$

Consider first the solution to the least square problem (28). From (7) the least square solution is given by

$$\mathbf{x} = (\mathbf{H}^*\mathbf{H} + \lambda\mathbf{I})^{-1}\mathbf{H}^*\,\mathbf{y}. \tag{30}$$

When $\mathbf{H}$ is a large matrix, we generally wish to avoid computing the solution (30) directly. In fact, we wish to avoid storing $\mathbf{H}$ in memory at all. However, note that the convolution matrix $\mathbf{H}$ in (27) is banded with $L$ non-zero diagonals. That is, $\mathbf{H}$ is a 'sparse banded' matrix. For such matrices, a sparse system solver can be used; i.e., an algorithm specifically designed to solve large systems of equations where the system matrix is a sparse matrix. Therefore, when $h$ is a short sequence, the least square solution (30) can be found efficiently using a solver for sparse banded linear

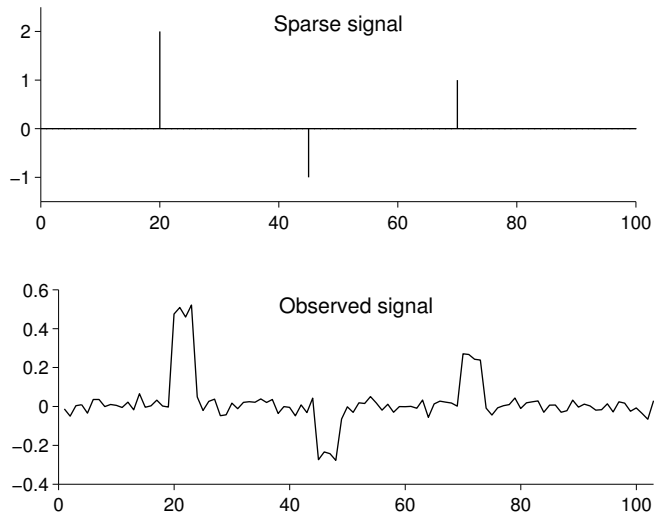


Figure 7: (a) Sparse signal and (b) observed signal $y$.

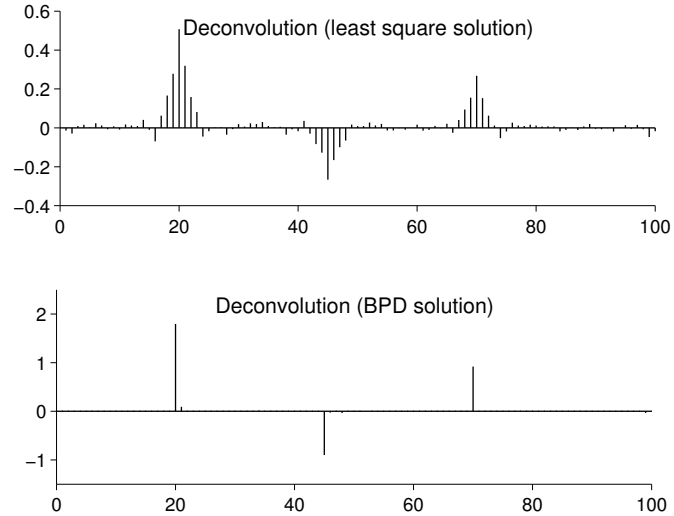


Figure 8: Deconvolution of observed signal in Fig. 7b. (a) Least square solution. (b) Basis pursuit denoising solution.

equations. In addition, when using such a solver, it is not necessary to store the matrix $\mathbf{H}$ in its entirety, only the non-zero diagonals.

To compute the solution to the basis pursuit denoising problem (29), we can use the iterative algorithm SALSA. This algorithm can also take advantage of solvers for sparse banded systems, so as to avoid the excessive computational cost and execution time of solving general large system of equations.

**Example**

A sparse signal is illustrated in Fig. 7a. This signal is convolved by the 4-point moving average filter

$$h(n) = \begin{cases} \frac{1}{4} & n = 0, 1, 2, 3 \\ 0 & \text{otherwise} \end{cases}$$

and noise is added to obtain the observed data $\mathbf{y}$ illustrated in Fig. 7b.

The least square and BPD solutions are illustrated in Fig. 8. In each case, the solution depends heavily on the parameter $\lambda$. In these two examples, we have chosen $\lambda$ manually so as to get as good a solution visually as we can. It is clear that the BPD solution is more accurate.

It is important to note that, if the original signal is not sparse, then the BPD method will not work. Its success depends on the original signal being sparse!

### 4.1 Exercises

1. Reproduce the example (or similar). Show the least square and BPD solutions for different values of $\lambda$ to illustrate how the solutions depend on $\lambda$. Comment on your observations.

## 5 Missing data estimation using BP

Due to data transmission errors or acquisition errors, some samples of a signal may be lost. In order to conceal these errors, the missing samples must be filled in with suitable values. Filling in missing values in order to conceal errors is called *error concealment* [26]. In some applications, part of a signal or image is intentionally deleted, for example in image editing so as to alter the image, or removing corrupted samples of an audio signal.

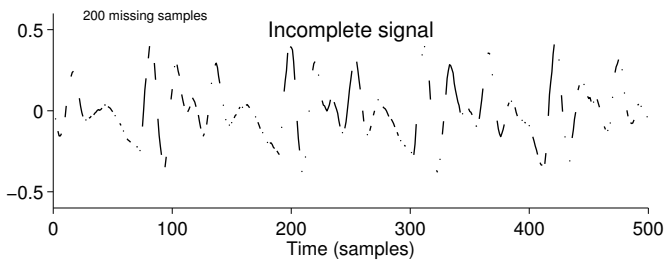


Figure 9: Signal with missing samples.

The samples or pixels should be convincingly filled in according to the surrounding area. In this case, the problem of filling in missing samples/pixels is often called *inpainting* [7]. Error concealment and inpainting both refer to filling in missing samples/pixels of a signal/image.

Several methods have been developed for filling in missing data, for example [16,17,19,20]. In this section, we will describe a method based on sparse signal representations. The example below involves one-dimensional signals, but the method can also be used for images, video, and other multidimensional data.

Fig. 9 shows an example of signal with missing samples. The signal is a speech waveform 500 samples long, but 200 samples are missing. That is, 300 of the 500 samples are known. The problem is to fill in the missing 200 samples.

### 5.1 Problem statement

Let $\mathbf{x}$ be a signal of length $M$. But suppose only $K$ samples of $\mathbf{x}$ are observed, where $K < M$. The $K$-point incomplete signal $\mathbf{y}$ can be written as

$$\mathbf{y} = \mathbf{S}\mathbf{x} \tag{31}$$

where $\mathbf{S}$ is a 'selection' (or 'sampling') matrix of size $K \times M$. For example, if only the first, second and last elements of a 5-point signal $\mathbf{x}$ are observed, then the matrix $\mathbf{S}$ is given by:

$$\mathbf{S} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{32}$$

The vector $\mathbf{y}$ is of length $K$ ($\mathbf{y}$ is shorter vector than $\mathbf{x}$). It consists of the known samples of $\mathbf{x}$.

The problem can be stated as: Given the incomplete signal $\mathbf{y}$ and the matrix $\mathbf{S}$, find $\mathbf{x}$ such that $\mathbf{y} = \mathbf{S}\mathbf{x}$. Of course, there are infinitely many solutions. Below, the least square and basis pursuit solutions will be illustrated. As will be shown, these two solutions are very different.

**Properties of S:** The matrix $\mathbf{S}$ has the following properties which will be used below.

1. Note that
$$\mathbf{S}\mathbf{S}^t = \mathbf{I} \tag{33}$$
where $\mathbf{I}$ is an $K \times K$ identity matrix. For example, for $\mathbf{S}$ in (32) have
$$\mathbf{S}\mathbf{S}^t = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

2. Also
$$\mathbf{S}^t\mathbf{S} = \mathrm{diag}(\mathbf{s}) \tag{34}$$
where the notation $\mathrm{diag}(\mathbf{s})$ denotes the diagonal matrix with $\mathbf{s}$ along the diagonal. For example, with $\mathbf{S}$ in (32) we have
$$\mathbf{S}^t\mathbf{S} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$
which we can write as
$$\mathbf{S}^t\mathbf{S} = \mathrm{diag}([1, \quad 1, \quad 0, \quad 0, \quad 1]).$$

3. Note that $\mathbf{S}^t\mathbf{y}$ has the effect of setting the missing samples to zero. For example, with $\mathbf{S}$ in (32) we have
$$\mathbf{S}^t\mathbf{y} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} y(0) \\ y(1) \\ y(2) \end{bmatrix} = \begin{bmatrix} y(0) \\ y(1) \\ 0 \\ 0 \\ y(2) \end{bmatrix}. \tag{35}$$

**Sparsity Model:** Suppose the signal $\mathbf{x}$ has a sparse representation with respect to $\mathbf{A}$, meaning that $\mathbf{x}$ can be represented as

$$\mathbf{x} = \mathbf{A}\mathbf{c} \tag{36}$$

where $\mathbf{c}$ is a sparse coefficient vector of length $N$ with $M \leq N$, and $\mathbf{A}$ is a matrix of size $M \times N$.

The incomplete signal $\mathbf{y}$ can then be written as

$$\begin{aligned} \mathbf{y} &= \mathbf{S}\mathbf{x} && \text{from (31)} && (37a) \\ &= \mathbf{S}\mathbf{A}\mathbf{c} && \text{from (36).} && (37b) \end{aligned}$$

Therefore, if we can find a vector $\mathbf{c}$ satisfying

$$\mathbf{y} = \mathbf{S}\mathbf{A}\mathbf{c} \tag{38}$$

then we can create an estimate $\hat{\mathbf{x}}$ of $\mathbf{x}$ by setting

$$\hat{\mathbf{x}} = \mathbf{A}\mathbf{c}. \tag{39}$$

From (38) it is clear that $\mathbf{S}\hat{\mathbf{x}} = \mathbf{y}$. That is, $\hat{\mathbf{x}}$ agrees with the known samples of $\mathbf{x}$. The signal $\hat{\mathbf{x}}$ is the same length as $\mathbf{x}$.

Note that $\mathbf{y}$ is shorter than the coefficient vector $\mathbf{c}$, so there are infinitely many solutions to (38). ($\mathbf{y}$ is of length $K$, and $\mathbf{c}$ is of length $N$.) Any vector $\mathbf{c}$ satisfying $\mathbf{y} = \mathbf{S}\mathbf{A}\mathbf{c}$ can be considered a valid set of coefficients. To find a particular solution we can minimize either $\|\mathbf{c}\|_2^2$ or $\|\mathbf{c}\|_1$. That is, we can solve either the least square problem:

$$\begin{aligned} &\underset{\mathbf{c}}{\operatorname{argmin}} \|\mathbf{c}\|_2^2 && (40a) \\ &\text{such that } \mathbf{y} = \mathbf{S}\mathbf{A}\mathbf{c} && (40b) \end{aligned}$$

or the basis pursuit problem:

$$\begin{aligned} &\underset{\mathbf{c}}{\operatorname{argmin}} \|\mathbf{c}\|_1 && (41a) \\ &\text{such that } \mathbf{y} = \mathbf{S}\mathbf{A}\mathbf{c}. && (41b) \end{aligned}$$

The two solutions are very different, as illustrated below.

Let us assume that $\mathbf{A}$ satisfies (10),

$$\mathbf{A}\mathbf{A}^* = p\mathbf{I}, \tag{42}$$

for some positive real number $p$.

Consider first the solution to the least square problem. From Section 1.2, the solution to least square problem (40) is given by

$$\begin{aligned} \mathbf{c} &= (\mathbf{S}\mathbf{A})^*((\mathbf{S}\mathbf{A})(\mathbf{S}\mathbf{A})^*)^{-1}\,\mathbf{y} && (43) \\ &= \mathbf{A}^*\mathbf{S}^t(\mathbf{S}\mathbf{A}\mathbf{A}^*\mathbf{S}^t)^{-1}\,\mathbf{y} && (44) \\ &= \mathbf{A}^*\mathbf{S}^t(p\,\mathbf{S}\mathbf{S}^t)^{-1}\,\mathbf{y} \qquad \text{using (10)} && (45) \\ &= \mathbf{A}^*\mathbf{S}^t(p\,\mathbf{I})^{-1}\,\mathbf{y} \qquad \text{using (33)} && (46) \\ &= \frac{1}{p}\mathbf{A}^*\mathbf{S}^t\,\mathbf{y} && (47) \end{aligned}$$

Hence, the least square estimate $\hat{\mathbf{x}}$ is given by

$$\begin{aligned} \hat{\mathbf{x}} &= \mathbf{A}\mathbf{c} && (48) \\ &= \frac{1}{p}\mathbf{A}\mathbf{A}^*\mathbf{S}^t\,\mathbf{y} \quad \text{using (47)} && (49) \\ &= \mathbf{S}^t\,\mathbf{y} \qquad \text{using (10).} && (50) \end{aligned}$$

But the estimate $\hat{\mathbf{x}} = \mathbf{S}^t\,\mathbf{y}$ consists of setting all the missing values to zero! See (35). No real estimation of the missing values has been achieved. The least square solution is of no use here.

Unlike the least square problem, the solution to the basis pursuit problem (41) can not be found in explicit form. It can be found only by running an iterative numerical algorithm. However, as illustrated in the following example, the BP solution can convincingly fill in the missing samples.

**Example**

Consider the problem of filling in the 200 missing samples of the length 500 signal $\mathbf{x}$ illustrated in Fig. 9. The number of known samples is $K = 300$, the length of the signal to be estimated is $M = 500$. Short segments of speech can be sparsely represented using the DFT; therefore we set $\mathbf{A}$ equal to the $M \times N$ matrix defined in (15) where we use $N = 1024$. Hence the coefficient vector $\mathbf{c}$ is of length 1024. To solve the BP problem (41), we have used 100 iterations of a SALSA algorithm.

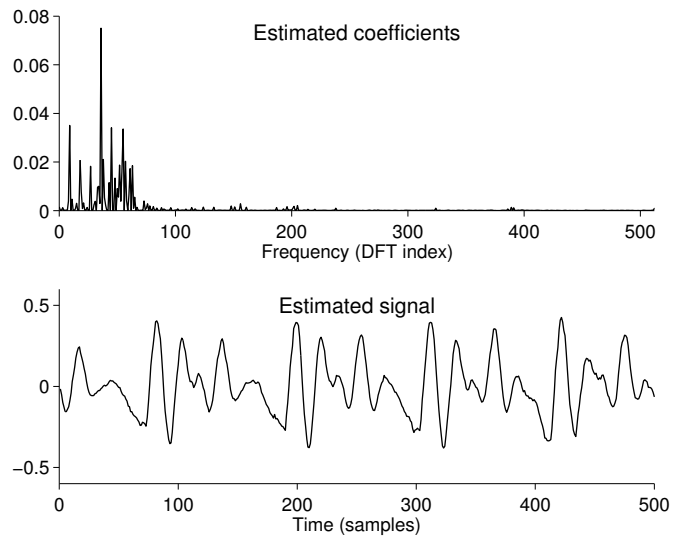


Figure 10: Estimation of missing signal values using basis pursuit.

The coefficient vector $\mathbf{c}$ obtained by solving (41) is illustrated in Fig. 10. (Only the first 512 coefficients are shown because $\mathbf{c}$ is symmetric.) The estimated signal $\hat{\mathbf{x}} = \mathbf{A}\mathbf{c}$ is also illustrated. The missing samples have been filled in quite accurately.

Unlike the least square solution, the BP solution fills in missing samples in a suitable way. But it should be emphasized that the effectiveness of the BP approach depends on the coefficients of the signal being *sparse* (or approximately sparse). For short speech segments, the Fourier transform can be used for sparse representation. Other types of signals may require other transforms.

### 5.2 Exercises

1. Reproduce the example (or similar). Show the least square and BP solutions for different numbers of missing samples. Comment on your observations.

## 6 Signal component separation using BP

When a signal contains components with distinct properties, it is sometimes useful to separate the signal into its components. If the components occupy distinct frequency bands, then the components can be separated using conventional LTI filters. However, if the components are not separated in frequency (or any other transform variable), then the separation of the components is more challenging. For example, a measured signal may contain both ($i$) transients of unspecified shape and ($ii$) sustained oscillations [21]; in order to perform frequency analysis, it can be beneficial to first remove the transients. But if the transient and oscillatory signal components overlap in frequency, then they can not be reliably separated using linear filters. It turns out that sparse representations can be used in this and other signal separation problems [3, 23, 24].

The use of sparsity to separate components of a signal has been called 'morphological component analysis' (MCA) [23, 24], the idea being that the components have different *shape* (morphology).

**Problem statement**

It is desired that the observed signal $\mathbf{y}$ be written as the sum of two components, namely,

$$\mathbf{y} = \mathbf{y}_1 + \mathbf{y}_2 \tag{51}$$

where $\mathbf{y}_1$ and $\mathbf{y}_2$ are to be determined. Of course, there are infinitely many ways to solve this problem. For example, one may set $\mathbf{y}_1 = \mathbf{y}$ and $\mathbf{y}_2 = \mathbf{0}$. Or one may set $\mathbf{y}_1$ arbitrarily and set $\mathbf{y}_2 = \mathbf{y} - \mathbf{y}_1$. However, it is assumed that each of the two components have particular properties and that they are distinct from one another. The MCA approach assumes that each of the two components is best described using distinct transforms $\mathbf{A}_1$ and $\mathbf{A}_2$, and aims to find the respective coefficients $\mathbf{c}_1$ and $\mathbf{c}_2$ such that

$$\mathbf{y}_1 = \mathbf{A}_1\mathbf{c}_1, \qquad \mathbf{y}_2 = \mathbf{A}_2\mathbf{c}_2. \tag{52}$$

Therefore, instead of finding $\mathbf{y}_1$ and $\mathbf{y}_2$ such that $\mathbf{y} = \mathbf{y}_1 + \mathbf{y}_2$, the MCA method finds $\mathbf{c}_1$ and $\mathbf{c}_2$ such that

$$\mathbf{y} = \mathbf{A}_1\mathbf{c}_1 + \mathbf{A}_2\mathbf{c}_2. \tag{53}$$

Or course, there are infinitely many solutions to this problem as well. To find a particular solution, we can use either least squares or basis pursuit. The least squares problem is:

$$\underset{\mathbf{c}_1,\,\mathbf{c}_2}{\operatorname{argmin}}\ \lambda_1 \|\mathbf{c}_1\|_2^2 + \lambda_2 \|\mathbf{c}_2\|_2^2 \tag{54a}$$

$$\text{such that } \mathbf{y} = \mathbf{A}_1\mathbf{c}_1 + \mathbf{A}_2\mathbf{c}_2. \tag{54b}$$

The 'dual basis pursuit' problem is:

$$\underset{\mathbf{c}_1,\,\mathbf{c}_2}{\operatorname{argmin}}\ \lambda_1 \|\mathbf{c}_1\|_1 + \lambda_2 \|\mathbf{c}_2\|_1 \tag{55a}$$

$$\text{such that } \mathbf{y} = \mathbf{A}_1\mathbf{c}_1 + \mathbf{A}_2\mathbf{c}_2. \tag{55b}$$

We call (55) 'dual' BP because of there being two sets of coefficients to be found. However, this problem is equivalent to basis pursuit.

Once $\mathbf{c}_1$ and $\mathbf{c}_2$ are obtained (using either least squares or dual BP), we set:

$$\mathbf{y}_1 = \mathbf{A}_1\,\mathbf{c}_1, \qquad \mathbf{y}_2 = \mathbf{A}_2\,\mathbf{c}_2. \tag{56}$$

Consider first the solution to the least square problem. The solution to the least square problem (54) is given by

$$\mu = \left[\frac{1}{\lambda_1}\mathbf{A}_1\mathbf{A}_1^* + \frac{1}{\lambda_2}\mathbf{A}_2\mathbf{A}_2^*\right]^{-1}\mathbf{y} \tag{57a}$$

$$\mathbf{c}_1 = \frac{1}{\lambda_1}\,\mathbf{A}_1^*\,\mu \tag{57b}$$

$$\mathbf{c}_2 = \frac{1}{\lambda_2}\,\mathbf{A}_2^*\,\mu. \tag{57c}$$

If the columns of $\mathbf{A}_1$ and $\mathbf{A}_2$ both form Parseval frames, i.e.

$$\mathbf{A}_1\mathbf{A}_1^* = p_1\mathbf{I}, \qquad \mathbf{A}_2\mathbf{A}_2^* = p_2\mathbf{I}, \tag{58}$$

then the least square solution (57) is given explicitly by

$$\mu = \left(\frac{p_1}{\lambda_1} + \frac{p_2}{\lambda_2}\right)^{-1}\mathbf{y} \tag{59a}$$

$$\mathbf{c}_1 = \frac{1}{\lambda_1}\left(\frac{p_1}{\lambda_1} + \frac{p_2}{\lambda_2}\right)^{-1}\mathbf{A}_1^*\,\mathbf{y} \tag{59b}$$

$$\mathbf{c}_2 = \frac{1}{\lambda_2}\left(\frac{p_1}{\lambda_1} + \frac{p_2}{\lambda_2}\right)^{-1}\mathbf{A}_2^*\,\mathbf{y} \tag{59c}$$

and from (56) the calculated components are

$$\mathbf{y}_1 = \frac{p_1}{\lambda_1}\left(\frac{p_1}{\lambda_1} + \frac{p_2}{\lambda_2}\right)^{-1}\mathbf{y} \tag{60a}$$

$$\mathbf{y}_2 = \frac{p_2}{\lambda_2}\left(\frac{p_1}{\lambda_1} + \frac{p_2}{\lambda_2}\right)^{-1}\mathbf{y}. \tag{60b}$$

But these are only *scaled* versions of the data $\mathbf{y}$. No real signal separation is achieved. The least square solution is of no use here.

Now consider the solution to the dual basis problem (55). It can not be found in explicit form; however, it can be obtained via a numerical algorithm. As illustrated in the following examples, the BP solution can separate a signal into distinct components, unlike the least square solution. This is true, provided that $\mathbf{y}_1$ has a sparse representation using $\mathbf{A}_1$, and that $\mathbf{y}_2$ has a sparse representation using $\mathbf{A}_2$.

**Example 1**

This first example is very simple; however, it clearly illustrates the method. Consider the 150-point signal in Fig. 11. It consists of a few impulses (spikes) and a few sinusoids added together. The goal is to separate the signal into two component signals: one component containing only impulses, the other component containing only sinusoids; such that, the sum of the two components equals the original signal.

To use the dual BP problem (55), we need to specify the two transforms $\mathbf{A}_1$ and $\mathbf{A}_2$. We set $\mathbf{A}_1$ to be the identity matrix. We set $\mathbf{A}_2$ to be the overcomplete DFT (15) with size $150 \times 256$. Hence the coefficient vector $\mathbf{c}_1$ is of length 150, and $\mathbf{c}_2$ is of length 256. The coefficients $\mathbf{c}_1$ and $\mathbf{c}_2$ obtained by solving the dual BP problem (55) are shown in Fig. 12. It can be seen that the coefficients are indeed sparse. (The coefficients $\mathbf{c}_2$ are complex valued, so the figure shows the modulus of these coefficients.) Using the coefficients $\mathbf{c}_1$ and $\mathbf{c}_2$, the components $\mathbf{y}_1$ and $\mathbf{y}_2$ are obtained by (56). The resulting components are illustrated in Fig. 12. It is clear that the dual BP method successfully separates the signal into two district components: $\mathbf{y}_1$ contains the impulses, and $\mathbf{y}_2$ contains the sinusoids.

In this illustrative example, the signal is especially simple, and the separation could probably have been accomplished with a simpler method.

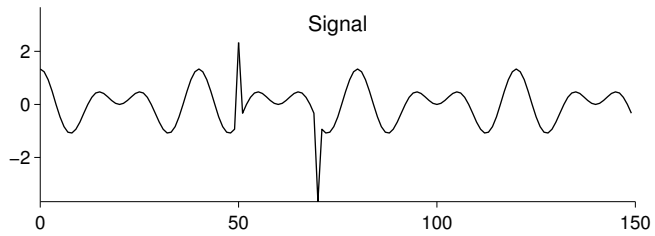


Figure 11: Signal composed of two distinct components.

The impulses are quite easily identified. However, in the next example, neither component can be so easily identified.

**Example 2**

An important property of a voiced speech waveform are the formants (prominent sustained frequencies). Formants are determined by the position of the tongue and lips, etc. (the vocal tract); and can be used to distinguish one vowel from another. Another important property of voiced speech is the pitch frequency, which is determined by the rate at which the vocal chords vibrate. The spectrum of a voiced speech waveform often contains many harmonic peaks at multiples of the pitch frequency. The pitch frequency and formants carry distinct information in the speech waveform; and in speech processing, they are estimated using different algorithms.

A speech signal and its spectrogram is shown in Fig. 13. The speech is sampled at 16K samples/second. Both formants and harmonics of the pitch frequency are clearly visible in the spectrogram: harmonics as fine horizontally oriented ridges, formants as broader darker bands.

Roughly, the formants and pitch harmonics can be separated to some degree using dual basis pursuit, provided appropriate transforms $\mathbf{A}_1$ and $\mathbf{A}_2$ are employed. We set each of the two transforms to be a short-time Fourier transform (STFT), but implemented with windows of differing lengths. For $\mathbf{A}_1$ we use a short window 32 samples (2 ms) and for $\mathbf{A}_2$ we use a longer window of 512 samples (32 ms). The result of dual BP is illustrated in Fig. 14. The arithmetic sum of the two components is equal to the original speech waveform, as constrained in the dual BP problem formulation. As the figure illustrates, the component $\mathbf{y}_1$ consists of brief

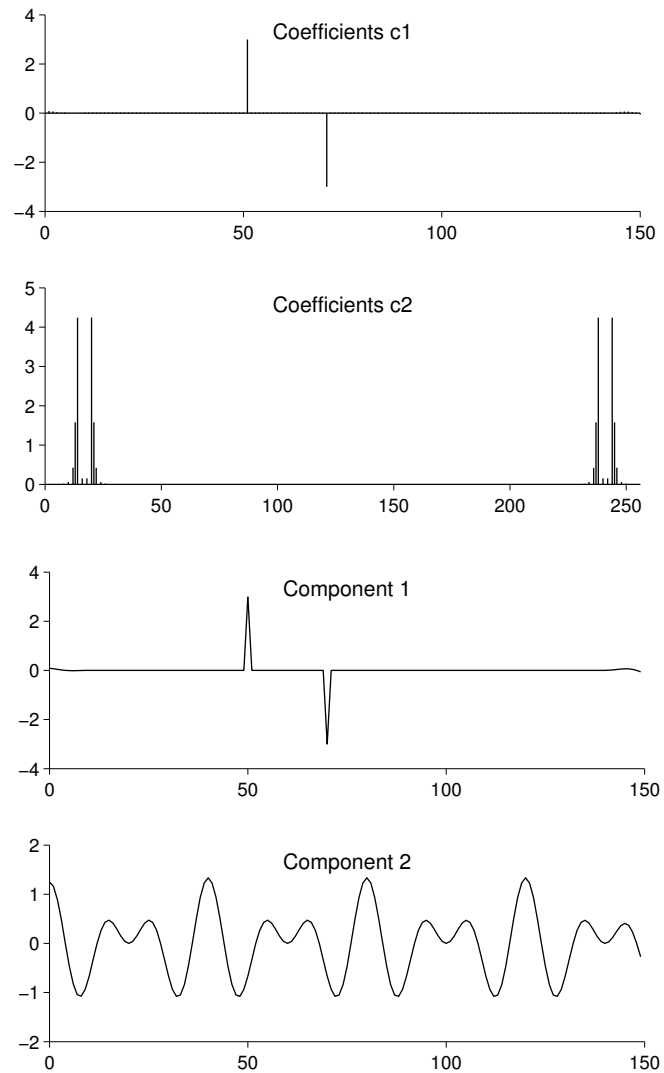


Figure 12: Signal separation using dual BP (Example 1). The two components add to give the signal illustrated in Fig. 11.

complex pulses, each shorter than the pitch period; the component $\mathbf{y}_2$ consists of sustained oscillations that are of longer duration. The sustained oscillations visible in $\mathbf{y}_2$ constitute the resonant frequencies of the vocal tract (formants). From the spectrogram of $\mathbf{y}_2$, illustrated in Fig. 15, it can be noted that the pitch harmonics have been substantially attenuated. This further illustrates that, to some degree, the pitch harmonics and formants have been separated into the components $\mathbf{y}_1$ and $\mathbf{y}_2$.

Note that the separation of formants and pitch harmonics is achieved even though the two components overlap in time and frequency. The dual BP procedure, with STFT of short and long windows, achieves this separation through sparse representation.

### 6.1 Exercises

1. Prove (57).

2. When $\mathbf{A}_1$ and $\mathbf{A}_2$ satisfy (10), prove (59a) and (60a).

3. Reproduce the examples (or similar). Show the dual BP solution for different parameters $\lambda_1$ and $\lambda_2$. Comment on your observations.

## 7 Conclusion

These notes describe the formulation of several problems. In each case, the least square solution is compared with the solution obtained using $\ell_1$ norm minimization (either BP or BPD).

1. *Signal representation* using basis pursuit (BP) is illustrated in Section 2. In the example, a sparse set of Fourier coefficients are obtained which do not exhibit the leakage phenomenon of the least square solution. For other types of data, it can be more suitable to use another transform, such as the short-time Fourier transform (STFT) or wavelet transforms [10], etc.

2. *Signal denoising* using basis pursuit denoising (BPD) is illustrated in Section 3. In the example, BPD is used to obtain a sparse set of Fourier coefficients that approximates the noisy signal. Denoising is achieved because the noise-free signal of interest has a sparse representation with respect to the Fourier transform. For other types of the

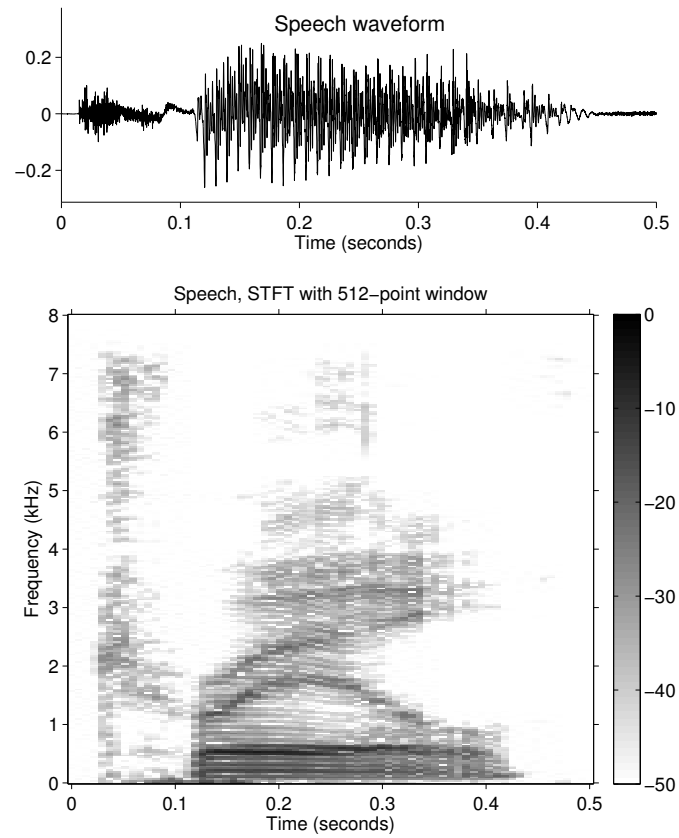


Figure 13: A speech waveform and its spectrogram.

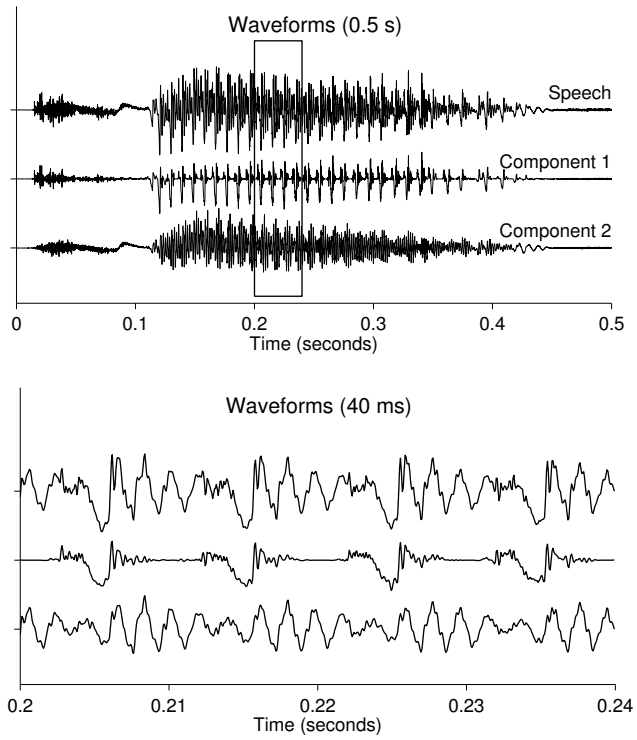


Figure 14: Speech signal decomposition using dual BP and STFT with short and long windows. The two components add to give the specch waveform illustrated in Fig. 13.

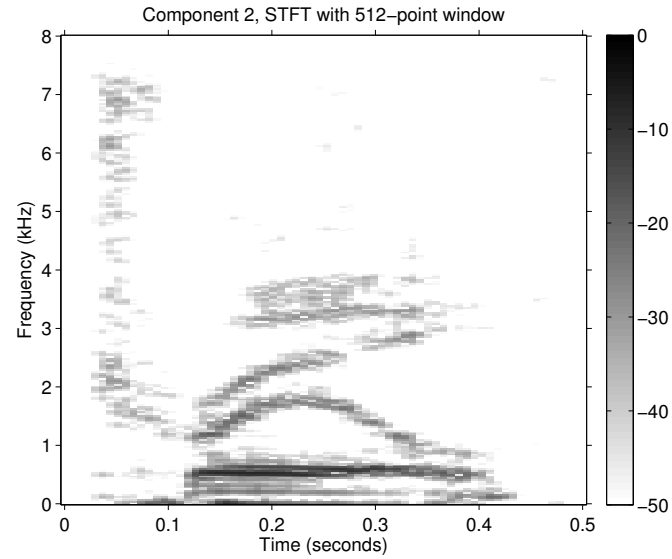


Figure 15: Spectrogram of component 2 in Fig. 14. Compared with Fig. 13 the pitch harmonics are reduced in intensity, but the formants are well preserved.

data, it can be more appropriate to use other transforms. A transform should be chosen that admits a sparse representation of the underlying signal of interest. It was also shown that the least square solution achieves no denoising, only a constant scaling of the noisy data.

3. *Deconvolution* using BPD is illustrated in Section 4. When the signal to be recovered is sparse, then the use of BPD can provide a more accurate solution than least squares. This approach can also be used when the signal to be recovered is itself not sparse, but admits a sparse representation with respect to a known transform.

4. *Missing data estimation* using BP is illustrated in Section 5. When the signal to be recovered in its entirety admits a sparse representation with respect to a known transform, then BP exploits this sparsity so as to fill in the missing data. If the data not only has missing samples but is also noisy, then BPD should be used in place of BP. It was shown that the least square solution consists of merely filling in the

missing data with zeros.

5. *Signal component separation* using BP is illustrated in Section 6. When each of the signal components admits a sparse representation with respect to known transforms, and when the transforms are sufficiently distinct, then the signal components can be isolated using sparse representations. If the data is not only a mixture of two components but is also noisy, then BPD should be used in place of BPD. It was shown that the least square solution achieves no separation; only a constant scaling of the observed mixed signal.

In each of these examples, the use of the $\ell_2$ norm in the problem formulation led to solutions that were unsatisfactory. Using the $\ell_1$ norm instead of the $\ell_2$ norm as in BP and BPD, lead to successful solutions. It should be emphasized that this is possible because (and only when) the signal to be recovered is either sparse itself or admits a sparse representation with respect to a known transform. Hence one problem in using sparsity in signal processing is to identity (or create) suitable transforms for different type of signals. Numerous specialized transforms have developed, especially for multidimensional signals [18].

In addition to the signal processing problems described in these notes, sparsity-based methods are central to *compressed sensing* [4].

These notes have intentionally avoided describing how to solve the BP and BPD problems, which can only be solved using iterative numerical algorithms. The derivation of algorithms for solving these problems will be described in separate notes.

## References


[1] M. V. Afonso, J. M. Bioucas-Dias, and M. A. T. Figueiredo. Fast image recovery using variable splitting and constrained optimization. *IEEE Trans. Image Process.*, 19(9):2345–2356, September 2010.

[2] M. V. Afonso, J. M. Bioucas-Dias, and M. A. T. Figueiredo. An augmented Lagrangian approach to the constrained optimization formulation of imaging inverse problems. *IEEE Trans. Image Process.*, 20(3):681–695, March 2011.

[3] J.-F. Aujol, G. Aubert, L. Blanc-Féraud, and A. Chambolle. Image decomposition into a bounded variation component and an oscillating component. *J. Math. Imag. Vis.*, 22:71–88, 2005.

[4] R. G. Baraniuk. Compressive sensing [lecture notes]. *IEEE Signal Processing Magazine*, 24(4):118–121, July 2007.

[5] R. G. Baraniuk, E. Candes, M. Elad, and Y. Ma, editors. Special issue on applications of sparse representation and compressive sensing. *Proc. IEEE*, 98(6), June 2010.

[6] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imag. Sci.*, 2(1):183–202, 2009.

[7] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher. Simultaneous structure and texture image inpainting. *IEEE Trans. on Image Processing*, 12(8):882–889, August 2003.

[8] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[9] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[10] C. S. Burrus, R. A. Gopinath, and H. Guo. *Introduction to Wavelets and Wavelet Transforms*. Prentice Hall, 1997.

[11] S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM J. Sci. Comput.*, 20(1):33–61, 1998.

[12] P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In H. H. Bauschke, R. Burachik, P. L. Combettes, V. Elser, D. R. Luke, and H. Wolkowicz, editors, *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*. Springer-Verlag (New York), 2010.

[13] I. Daubechies, M. Defriese, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Commun. Pure Appl. Math*, LVII:1413–1457, 2004.

[14] M. Figueiredo and R. Nowak. An EM algorithm for wavelet-based image restoration. *IEEE Trans. Image Process.*, 12(8):906–916, August 2003.

[15] R. Gribonval and M. Nielsen, editors. Special issue on sparse approximations in signal and image processing. *Signal Processing*, 86(3):415–638, March 2006.

[16] O. G. Guleryuz. Nonlinear approximation based image recovery using adaptive sparse reconstructions and iterated denoising — Part I: Theory. *IEEE Trans. on Image Processing*, 15(3):539–554, March 2006.

[17] O. G. Guleryuz. Nonlinear approximation based image recovery using adaptive sparse reconstructions and iterated denoising — Part II: Adaptive algorithms. *IEEE Trans. on Image Processing*, 15(3):555–571, March 2006.

[18] L. Jacques, L. Duval, C. Chaux, and G. Peyré. A panorama on multiscale geometric representations, intertwining spatial, directional and frequency selectivity. *Signal Processing*, 91(12):2699–2730, December 2011. Special issue on advances in multirate filter bank structures and multiscale representations.

[19] A. Kaup, K. Meisinger, and T. Aach. Frequency selective signal extrapolation with applications to error concealment in image communication. *Int. J. of Electronics and Communications*, 59:147–156, 2005.

[20] K. Meisinger and A. Kaup. Spatiotemporal selective extrapolation for 3-D signals and its applications in video communications. *IEEE Trans. on Image Processing*, 16(9):2348–2360, September 2007.

[21] I. W. Selesnick. Resonance-based signal decomposition: A new sparsity-enabled signal analysis method. *Signal Processing*, 91(12):2793 – 2809, 2011.

[22] *Workshop: Signal Processing with Adaptive Sparse Structured Representations (SPARS11)*, Edinburgh, June 2011. http://ecos.maths.ed.ac.uk/SPARS11/.

[23] J.-L. Starck, M. Elad, and D. Donoho. Redundant multiscale transforms and their application for morphological component analysis. *Advances in Imaging and Electron Physics*, 132:287–348, 2004.

[24] J.-L. Starck, M. Elad, and D. Donoho. Image decomposition via the combination of sparse representation and a variational approach. *IEEE Trans. Image Process.*, 14(10):1570–1582, October 2005.

[25] R. Tibshirani. Regression shrinkage and selection via the lasso. *J. Roy. Statist. Soc., Ser. B*, 58(1):267–288, 1996.

[26] Y. Wang and Q. Zhu. Error control and concealment for video communication: A review. *Proc. IEEE*, 86(5):974–997, May 1998.