Introduction to Sparsity in Signal Processing

Ivan Selesnick

Polytechnic Institute of New York University

Brooklyn, New York

selesi@poly.edu

2012
Under-determined linear equations

Consider a system of under-determined system of equations

\[ y = Ax \] (1)

A : \( M \times N \) \hspace{1cm} M < N

y : length-\( M \)

x : length-N

\[ y = \begin{bmatrix} y(0) \\ \vdots \\ y(M-1) \end{bmatrix} \quad x = \begin{bmatrix} x(0) \\ \vdots \\ x(N-1) \end{bmatrix} \]

The system has more unknowns than equations.

The matrix \( A \) is wider than it is tall.

We assume that \( AA^* \) is invertible, therefore the system of equations (1) has infinitely many solutions.
We will use the $\ell_2$ and $\ell_1$ norms:

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

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

$\|x\|_2^2$, i.e. the sum of squares, is referred to as the ‘energy’ of $x$. 
Least squares

To solve $y = Ax$, it is common to minimize the energy of $x$.

$$\arg\min_x \|x\|_2^2$$ (4a)

such that $y = Ax$. (4b)

Solution to (4):

$$x = A^*(AA^*)^{-1}y.$$ (5)

When $y$ is noisy, don’t solve $y = Ax$ exactly. Instead, find approximate solution:

$$\arg\min_x \|y - Ax\|_2^2 + \lambda \|x\|_2^2.$$ (6)

Solution to (6):

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

Large scale systems $\rightarrow$ fast algorithms needed.
Sparse solutions

Another approach to solve \( y = Ax \):

\[
\arg\min_x \|x\|_1 \quad (8a)
\]

such that \( y = Ax \) \quad (8b)

Problem (8) is the basis pursuit (BP) problem.

When \( y \) is noisy, don’t solve \( y = Ax \) exactly. Instead, find approximate solution:

\[
\arg\min_x \|y - Ax\|_2^2 + \lambda \|x\|_1. \quad (9)
\]

Problem (9) is the basis pursuit denoising (BPD) problem.

The BP/BPD problems can not be solved in explicit form, only by iterative numerical algorithms.
Least squares & BP/BPD

Least squares and BP/BPD solutions are quite different. Why?

To minimize $\|x\|_2^2$ . . . the largest values of $x$ must be made small as they count much more than the smallest values.

$\Rightarrow$ least square solutions have many small values, as they are relatively unimportant $\Rightarrow$ least square solutions are not sparse.

Therefore, when it is known/expected that $x$ is sparse, use the $\ell_1$ norm; not the $\ell_2$ norm.
Algorithms for sparse solutions

Cost function:

- Non-differentiable
- Convex
- Large-scale

Algorithms:

- ‘Matrix-free’
- ISTA
- FISTA
- SALSA (ADMM)

and more...
Parseval frames

If \( \mathbf{A} \) satisfies
\[
\mathbf{A} \mathbf{A}^* = p \mathbf{I}
\] (10)
then the columns of \( \mathbf{A} \) are said to form a ‘Parseval frame’. \( \mathbf{A} \) can be rectangular.

Least squares solution (5) becomes
\[
x = \mathbf{A}^* (\mathbf{A} \mathbf{A}^*)^{-1} \mathbf{y} = \frac{1}{p} \mathbf{A}^* \mathbf{y} \quad (\mathbf{A} \mathbf{A}^* = p \mathbf{I})
\]
(11)
No matrix inversion needed.

Least square solution (7) becomes
\[
x = (\mathbf{A}^* \mathbf{A} + \lambda \mathbf{I})^{-1} \mathbf{A}^* \mathbf{y} = \frac{1}{\lambda + p} \mathbf{A}^* \mathbf{y} \quad (\mathbf{A} \mathbf{A}^* = p \mathbf{I})
\]
(12)
(Use the matrix inverse lemma.) No matrix inversion needed.

If \( \mathbf{A} \) satisfies (10), then it is very easy to find least square solutions. Some algorithms for BP/BPD also become computationally easier.
Example: Sparse Fourier coefficients using BP

The Fourier transform tells how to write the signal as a sum of sinusoids. But, it is not the only way.

Basis pursuit gives a sparse spectrum.

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

$$y(m) = \sum_{n=0}^{N-1} c(n) \exp\left(j \frac{2\pi}{N} mn\right), \quad 0 \leq m \leq M - 1$$

where $c(n)$ is a length-$N$ coefficient sequence, with $M \leq N$.

$$y = Ac$$

$$A_{m,n} = \exp\left(j \frac{2\pi}{N} mn\right), \quad 0 \leq m \leq M - 1, \quad 0 \leq n \leq N - 1$$

$c : \text{length-}N$

The coefficients $c(n)$ are frequency-domain (Fourier) coefficients.
Example: Sparse Fourier coefficients using BP

1. If $N = M$, then $A$ is the inverse $N$-point DFT matrix.
2. If $N > M$, then $A$ is the first $M$ rows of the inverse $N$-point DFT matrix.
   \[ \Rightarrow A \text{ or } A^* \text{ can be implemented efficiently using the FFT.} \]
   For example, in Matlab, $y = Ac$ is implemented as:

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

   Similarly, $A^* y$ can be obtained by zero-padding and computing the DFT.
   In Matlab, $c = A^* y$ is implemented as:

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

   \[ \Rightarrow \text{Matrix-free algorithms.} \]

3. Due to the orthogonality properties of complex sinusoids,

   \[ AA^* = NI_M \]  (16)
Example: Sparse Fourier coefficients using BP

When $N = M$, the coefficients $c$ satisfying $y = Ac$ are uniquely determined.

When $N > M$, the coefficients $c$ are not unique. Any vector $c$ satisfying $y = Ac$ can be considered a valid set of coefficients. To find a particular solution we can minimize either $\|c\|_2^2$ or $\|c\|_1$.

Least squares:

$$\arg\min_c \|c\|_2^2 \quad (17a)$$

such that $y = Ac \quad (17b)$

Basis pursuit:

$$\arg\min_c \|c\|_1 \quad (18a)$$

such that $y = Ac. \quad (18b)$

The two solutions can be quite different...
Example: Sparse Fourier coefficients using BP

Least square solution:

\[ c = A^* (A A^*)^{-1} y = \frac{1}{N} A^* y \]  

(least square solution)

\( A^* y \) is computed via:

1. zero-pad the length-\( M \) signal \( y \) to length-\( N \)
2. compute its DFT

BP solution: Compute using algorithm SALSA
Example: Sparse Fourier coefficients using BP

The BP solution does not exhibit the leakage phenomenon.
Example: Sparse Fourier coefficients using BP

Cost function history of algorithm for basis pursuit solution
Example: Denoising using BPD

Digital LTI filters are often used for noise reduction (denoising).

But if

- the noise and signal overlap in the frequency domain
  or
- the respective frequency bands are unknown,

then it is difficult to use LTI filters.

However, if the signal has sparse (or relatively sparse) Fourier coefficients, then BPD can be used for noise reduction.
Example: Denoising using BPD

Noisy speech signal $y(m)$:

$$y(m) = s(m) + w(m), \quad 0 \leq m \leq M - 1, \quad M = 500 \quad (19)$$

$s(m)$: noise-free speech signal

$w(m)$: noise sequence.
Example: Denoising using BPD

Assume the noise-free speech signal $s(n)$ has a sparse set of Fourier coefficients:

$$ y = Ac + w $$

$y$: noisy speech signal, length-$M$

$A$: $M \times N$ DFT matrix (15)

$c$: sparse Fourier coefficients, length-$N$

$w$: noise, length-$M$

As $y$ is noisy, find $c$ by solving the least square problem

$$ \arg\min_c \|y - Ac\|_2^2 + \lambda \|c\|_2^2 $$ \hspace{1cm} (20)

or the basis pursuit denoising (BPD) problem

$$ \arg\min_c \|y - Ac\|_2^2 + \lambda \|c\|_1. $$ \hspace{1cm} (21)

Once $c$ is found, an estimate of the speech signal is given by $\hat{s} = Ac$. 
Example: Denoising using BPD

Least square solution:

\[ c = (A^*A + \lambda I)^{-1}A^*y = \frac{1}{\lambda + N} A^*y \quad (AA^* = N I) \quad (22) \]

using (12) and (16).
⇒ least square estimate of the speech signal is

\[ \hat{s} = Ac = \frac{N}{\lambda + N} y \quad \text{(least square solution).} \]

But this is only a scaled version of the noisy signal!

No real filtering is achieved.
Example: Denoising using BPD

BPD solution:

obtained with algorithm SALSA. Effective noise reduction, unlike least squares!
Example: Deconvolution using BPD

If the signal of interest $x(m)$ is not only noisy but is also distorted by an LTI system with impulse response $h(m)$, then the available data $y(m)$ is

$$y(m) = (h \ast x)(m) + w(m)$$  \hspace{1cm} (23)$$

where ‘$\ast$’ 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.
Example: Deconvolution using BPD

\[ y = Hx + w \]  \hspace{1cm} (24)

- \( x \): length \( N \)
- \( h \): length \( L \)
- \( y \): length \( M = N + L - 1 \)

\[
H = \begin{bmatrix}
    h_0 & h_0 & h_0 \\
    h_1 & h_1 & h_1 \\
    h_2 & h_2 & h_2 \\
    h_2 & h_2 & h_2 \\
\end{bmatrix}
\]  \hspace{1cm} (25)

\( H \) is of size \( M \times N \) with \( M > N \) (because \( M = N + L - 1 \)).
Example: Deconvolution using BPD

Sparse signal convolved by the 4-point moving average filter

\[ h(n) = \begin{cases} 
\frac{1}{4} & n = 0, 1, 2, 3 \\
0 & \text{otherwise}
\end{cases} \]
Example: Deconvolution using BPD

Due to noise $w$, solve the least square problem

$$\arg\min_x \|y - Hx\|_2^2 + \lambda \|x\|_2^2$$

or the basis pursuit denoising (BPD) problem

$$\arg\min_x \|y - Hx\|_2^2 + \lambda \|x\|_1$$

Least square solution:

$$x = (H^*H + \lambda I)^{-1}H^*y.$$
Example: Deconvolution using BPD

The BPD solution, obtained using SALSA, is more faithful to original signal.
Due to data transmission/acquisition errors, some signal samples may be lost. Fill in missing values for *error concealment*.

Part of a signal or image may be intentionally deleted (image editing, etc). Convincingly fill in missing values according to the surrounding area to do *inpainting*.
Example: Filling in missing samples using BP

The incomplete signal $y$:

$$y = Sx$$  \hspace{1cm} (29)$$

$x$: length $M$
$y$: length $K < M$
$S$: ‘selection’ (or ‘sampling’) matrix of size $K \times M$.

For example, if only the first, second and last elements of a 5-point signal $x$ are observed, then the matrix $S$ is given by:

$$S = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \hspace{1cm} (30)$$

Problem: Given $y$ and $S$, find $x$ such that $y = Sx$

$\Rightarrow$ Underdetermined system, infinitely many solutions.

Least square and BP solutions are very different...
Example: Filling in missing samples using BP

Properties of $S$:

1. $SS^t = I$  \hspace{1cm} (31)
   
   where $I$ is an $K \times K$ identity matrix. For example, with $S$ in (30)
   
   $$SS^t = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$  

2. $S^t y$ sets the missing samples to zero.
   
   For example, with $S$ in (30)
   
   $$S^t y = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} y(0) \\ y(1) \\ y(2) \end{bmatrix} = \begin{bmatrix} y(0) \\ y(1) \\ 0 \\ 0 \\ y(2) \end{bmatrix}. \hspace{1cm} (32)$$
Example: Filling in missing samples using BP

Suppose $x$ has a sparse representation with respect to $A$, $x = Ac$ \hspace{1cm} (33)

c: sparse vector, length $N$, with $M \leq N$
A: size $M \times N$.

The incomplete signal $y$ can then be written as

$y = Sx$ \hspace{1cm} \text{from (50)} \hspace{1cm} (34a) \\
= SAc$ \hspace{1cm} \text{from (33)}. \hspace{1cm} (34b)

Therefore, if we can find a vector $c$ satisfying

$y = SAc$ 

(35)

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

$\hat{x} = Ac$. 

(36)

From (35), $S\hat{x} = y$. That is, $\hat{x}$ agrees with the known samples $y$. 

Example: Filling in missing samples using BP

Note that \( y \) is shorter than the coefficient vector \( c \), so there are infinitely many solutions to (35).

Any vector \( c \) satisfying \( y = S A c \) can be considered a valid set of coefficients.

To find a particular solution, solve the least square problem

\[
\begin{align*}
\text{argmin} \quad & \| c \|_2^2 \\
\text{such that} \quad & y = S A c
\end{align*}
\]  

(37a) (37b)

or the basis pursuit problem

\[
\begin{align*}
\text{argmin} \quad & \| c \|_1 \\
\text{such that} \quad & y = S A c.
\end{align*}
\]  

(38a) (38b)

The two solutions are very different...

Let us assume \( A \) satisfies

\[
A A^* = p I,
\]

for some positive real number \( p \).
Example: Filling in missing samples using BP

The least square solution is

\[ c = (SA)^\ast((SA)(SA)^\ast)^{-1}y \]  \hspace{1cm} (40)
\[ = A^\ast S^t(SAA^\ast S^t)^{-1}y \]  \hspace{1cm} (41)
\[ = A^\ast S^t(pSS^t)^{-1}y \quad \text{using (10)} \]  \hspace{1cm} (42)
\[ = A^\ast S^t(pI)^{-1}y \quad \text{using (31)} \]  \hspace{1cm} (43)
\[ = \frac{1}{p}A^\ast S^t y \]  \hspace{1cm} (44)

Hence, the least square estimate \( \hat{x} \) is given by

\[ \hat{x} = Ac \]  \hspace{1cm} (45)
\[ = \frac{1}{p}AA^\ast S^t y \quad \text{using (44)} \]  \hspace{1cm} (46)
\[ = S^t y \quad \text{using (10).} \]  \hspace{1cm} (47)

This estimate consists of setting all the missing values to zero! See (32).

No real estimation of the missing values has been achieved.

The least square solution is of no use here.
Example: Filling in missing samples using BP

Short segments of speech can be sparsely represented using the DFT; therefore we set $A$ equal to the $M \times N$ DFT (15) with $N = 1024$.

BP solution obtained using 100 iterations of a SALSA:

The missing samples have been filled in quite accurately.
Total Variation Denoising/Deconvolution

A signal is observed in additive noise,

\[ y = x + w. \]

*Total variation* denoising is suitable when it is known that the signal of interest has a sparse derivative.

\[
\text{argmin}_x \| y - x \|_2^2 + \lambda \sum_{n=2}^{N} |x(n) - x(n - 1)|. \tag{48}
\]

The total variation of \( x \) is defined as

\[
\text{TV}(x) := \sum_{n=2}^{N} |x(n) - x(n - 1)| = \| Dx \|_1
\]

\[
D = \begin{bmatrix}
-1 & 1 & 0 & \ldots & 0 \\
-1 & 1 & -1 & \ldots & 0 \\
\vdots & \ddots & \ddots & \ddots & \ddots \\
0 & \ldots & -1 & 1 & -1 \\
-1 & 1
\end{bmatrix}
\tag{49}
\]

Matrix \( D \) has size \((N - 1) \times N\).
Total Variation Denoising/Deconvolution

TV filtering preserves edges/jumps more accurately than least squares filtering.
Polynomial Smoothing of Time Series with Additive Step Discontinuities

Problem: estimate simultaneously a local polynomial signal and an approximately piecewise constant signal from a noisy additive mixture.\(^1\)

The given discrete-time data \(y(n)\) is modeled as

\[
y(n) = s(n) + x(n) + w(n), \quad n = 1, \ldots, N
\]  

(50)

where:

1. The signal \(s(n)\) is well approximated, on each interval of \(L\) samples, by a polynomial of degree \(d\), with \(d \ll L\).
2. \(x(n)\) is approximately piecewise constant.
3. \(w(n)\) is stationary white Gaussian noise.

Such data arises when a discrete event phenomenon is observed in the presence of a comparatively slow varying signal.

- Detection of virus particles using the Whispering Gallery Mode (WGM) biosensor (Arnold et. al.).
- Near infrared spectroscopic (NIRS) imaging systems (Graber et. al.).

\(^1\)MATLAB software available at http://eeweb.poly.edu/iselesni/patv/
Polynomial Smoothing of Time Series with Additive Step Discontinuities

**Variational Formulation:** Given noisy data $y$, find

$x$ (step signal)

$p$ (local polynomial signal)

by minimization:

$$
\arg\min_{a,x} \lambda \sum_{n=2}^{N} |x(n) - x(n-1)| + \sum_{n=1}^{N} |y(n) - p(n) - x(n)|^2
$$

where

$$p(n) = a_1 n + \cdots + a_d n^d.$$

The cost function is non-differentiable. We use ‘variable splitting’ and ADMM to derive an efficient iterative algorithm.
Observed noisy data: a low-frequency signal and with step-wise discontinuities (steps/jumps).

Goal: Simultaneous smoothing and change/jump detection.

Conventional low-pass filtering blurs discontinuities.
Nano-particle Detection: The Whispering Gallery Mode (WGM) biosensor, designed to detect individual nano-particles in real time, is based on detecting discrete changes in the resonance frequency of a microspherical cavity. The adsorption of a nano-particle onto the surface of a custom designed microsphere produces a theoretically predicted but minute change in the optical properties of the sphere [Arnold, 2003].
Nano-particle Detection:

Summary

1. Basis pursuit
2. Basis pursuit denoising
3. Sparse Fourier coefficients using BP
4. Denoising using BPD
5. Deconvolution using BPD
6. Filling in missing samples using BP
7. Total variation filtering
8. Simultaneous least-squares & sparsity