SPARSE SIGNAL RESTORATION

IVAN W. SELESNICK

1. Introduction

These notes describe an approach for the restoration of degraded signals using sparsity. This approach, which has become quite popular, is useful for numerous problems in signal processing: denoising, deconvolution, interpolation, super-resolution, declipping, etc [6].

We assume that the observed signal $\mathbf{y}$ can be written as

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}$$

where $\mathbf{x}$ is the signal of interest which we want to estimate, $\mathbf{n}$ is additive noise, and $\mathbf{H}$ is a matrix representing the observation processes. For example, if we observe a blurred version of $\mathbf{x}$ then $\mathbf{H}$ will be a convolution matrix.

The estimation of $\mathbf{x}$ from $\mathbf{y}$ can be viewed as a linear inverse problem. A standard approach to solve linear inverse problems is to define a suitable objective function $J(\mathbf{x})$ and to find the signal $\mathbf{x}$ minimizing $J(\mathbf{x})$. Generally, the chosen objective function is the sum of two terms:

$$J(\mathbf{x}) = D(\mathbf{y}, \mathbf{H}\mathbf{x}) + \lambda R(\mathbf{x})$$

where $D(\mathbf{y}, \mathbf{H}\mathbf{x})$ measures the discrepancy between $\mathbf{y}$ and $\mathbf{x}$ and $R(\mathbf{x})$ is a regularization term (or penalty function). The parameter λ is called the regularization parameter and is used to adjust the trade-off between the two terms; λ should be a positive value. On one hand, we want to find a signal $\mathbf{x}$ so that $\mathbf{H}\mathbf{x}$ is very similar to $\mathbf{y}$; that is, we want to find a signal $\mathbf{x}$ which is consistent with the observed data $\mathbf{y}$. For $D(\mathbf{y}, \mathbf{H}\mathbf{x})$, we will use the mean square error, namely

$$D(\mathbf{y}, \mathbf{H}\mathbf{x}) = \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2.$$

The notation $\|\mathbf{v}\|_2^2$ represents the sum of squares of the vector $\mathbf{v}$,

$$\|\mathbf{v}\|_2^2 := v_1^2 + v_2^2 + \cdots + v_N^2.$$

Minimizing this $D(\mathbf{y}, \mathbf{H}\mathbf{x})$ will give a signal $\mathbf{x}$ which is as consistent with $\mathbf{y}$ as possible according to the square error criterion. We could try to minimize $D(\mathbf{y}, \mathbf{H}\mathbf{x})$ by setting $\mathbf{x} = \mathbf{H}^{-1}\mathbf{y}$; however, $\mathbf{H}$ may not be invertible. Even if $\mathbf{H}$ were invertible, it may be very ill-conditioned in which case this solution amplifies the noise, sometimes so much that the solution is useless. The role of the regularization term $R(\mathbf{x})$ is exactly to address this problem. The regularizer $R(\mathbf{x})$ should be chosen so as to penalize undesirable/unwanted behaviour in $\mathbf{x}$. For example, if it is expected that $\mathbf{x}$ is a smooth signal, then $R(\mathbf{x})$ should be chosen to penalize non-smooth signals. For example one could set $R(\mathbf{x}) = \sum_n |x(n) - x(n-1)|$. (This $R(\mathbf{x})$ is called the 'total variation' of $\mathbf{x}$ and is zero only when $\mathbf{x}$ is a constant-valued signal. The more the signal $\mathbf{x}$ varies, the greater is its total variation.)

Date: September 24, 2009.

Last edit: April 25, 2014.

This lecture note is a Connexions module (m32168 at `http://cnx.org/`).

Cite as: I. Selesnick. Sparse Signal Restoration. Connexions, 2009. http://cnx.org/content/m32168/.

In these notes, it is assumed that the signal of interest, $\mathbf{x}$, is known to be *sparse*. That is, $\mathbf{x}$ has relatively few non-zero values. For example, $\mathbf{x}$ may consist of a few impulses and is otherwise zero. In this case, we should define $R(\mathbf{x})$ to be the number of non-zero values of $\mathbf{x}$. Unfortunately, with $R(\mathbf{x})$ defined as such, the objective function $J(\mathbf{x})$ is very difficult to minimize. First, this $R(\mathbf{x})$ is not differentiable. Second, and more importantly, this $R(\mathbf{x})$ is not a *convex* function of $\mathbf{x}$, and therefore $J(\mathbf{x})$ will have many local minima. In order to develop robust numerical algorithms for the minimization of $J(\mathbf{x})$, it is best that $J(\mathbf{x})$ be a convex function of $\mathbf{x}$. We would like a function that measures sparsity, but which is also convex. For this reason, when $\mathbf{x}$ is known to be sparse, the regularization function $R(\mathbf{x})$ is often chosen to be the ℓ_1-norm, which is defined as

$$\|\mathbf{x}\|_1 := |x_1| + |x_2| + \cdots + |x_N|.$$

Hence, the approach is to estimate $\mathbf{x}$ from $\mathbf{y}$ by minimizing the objective function

$$J(\mathbf{x}) = \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_1. \tag{1}$$

This is called an ℓ_1-norm regularized linear inverse problem.

The development of fast algorithm to minimize (1), and related functions, is an active research topic. An early and important algorithm is the iterated soft-thresholding algorithm (ISTA), also called the thresholded-Landweber (TL) algorithm. This algorithm was developed in [4, 9] for the purpose of wavelet-based signal restoration, but the algorithm in a more general form appeared earlier in the optimization literature, see [1]. Since the development of ISTA, other faster algorithms have been introduced for the minimization of (1), for example [1–3, 5, 7, 7, 8, 10–14] and others. Some of these algorithms can be used for the minimization of more general or related objective functions as well, not just (1).

These notes describe the derivation of the iterated soft-thresholding algorithm (ISTA). ISTA is a combination of the Landweber algorithm and soft-thresholding (so it is also called the thresholded-Landweber algorithm). The derivation below is based on majorization-minimization (a concept in optimization theory) and on ℓ_1-norm regularized denoising (which leads to soft-thresholding). The derivation uses linear algebra (positive definite matrices) and vector derivatives.

2. Majorization-Minimization

Majoriziation-minimization (MM) replaces a difficult minimization problem by a sequence of easier minimization problems. The MM approach generates a sequence of vectors $\mathbf{x}_k$, $k = 0, 1, 2, \dots$ which converge to desired solution. The MM approach is useful for the minimization of (1) because $J(\mathbf{x})$ can not be easily minimized. There is no formula for the vector $\mathbf{x}$ that minimizes $J(\mathbf{x})$.

The majoriziation-minimization idea can be described as follows. Suppose we have a vector $\mathbf{x}_k$, a 'guess' for the minimum of $J(\mathbf{x})$. Based on $\mathbf{x}_k$, we would like to find a new vector, $\mathbf{x}_{k+1}$ which further decreases $J(\mathbf{x})$; that is, we want to find $\mathbf{x}_{k+1}$ such that

$$J(\mathbf{x}_{k+1}) < J(\mathbf{x}_k).$$

The MM approach asks us first to choose a new function which majorizes $J(\mathbf{x})$ and, second, that we minimize the new function to get $\mathbf{x}_{k+1}$. MM puts some requirements on this new function, call it $G(\mathbf{x})$. We should choose $G(\mathbf{x})$ such that $G(\mathbf{x}) \geq J(\mathbf{x})$ for all $\mathbf{x}$ (that is what it means that $G(\mathbf{x})$ majorizes $J(\mathbf{x})$). In addition $G(\mathbf{x})$ should equal $J(\mathbf{x})$ at $\mathbf{x}_k$. We find $\mathbf{x}_{k+1}$ by minimizing $G(\mathbf{x})$. For this method to be useful, we should choose $G(\mathbf{x})$ to be a function we can minimize easily. The function $G(\mathbf{x})$ will be different at each iteration, so we denote it $G_k(\mathbf{x})$.

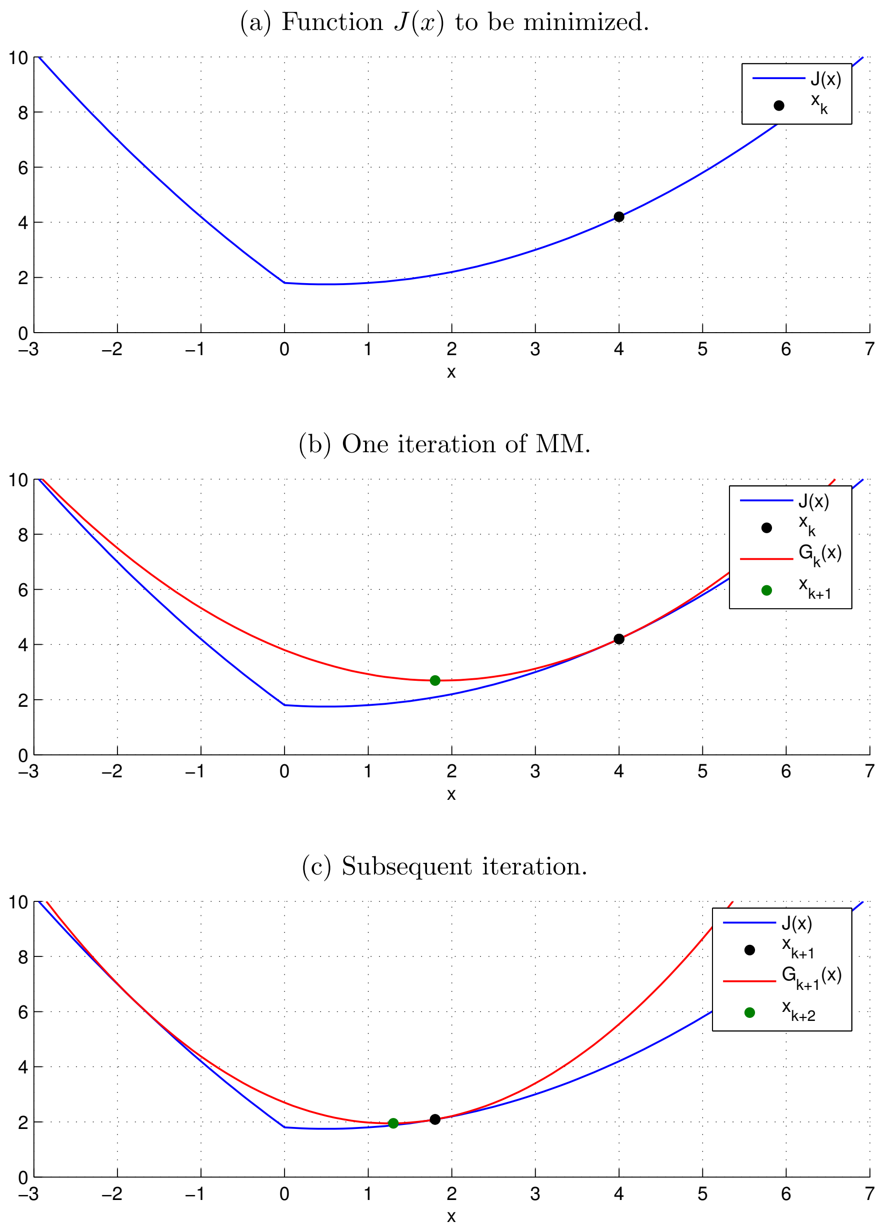

FIGURE 1. Illustration of majorization-minimization (MM) procedure. Plot (a) illustrates the function to be minimized, $J(x)$, and a 'guess' of the minimizer, x_k. Plot (b) illustrates the majorizing function $G_k(x)$ which coincides with $J(x)$ at $x = x_k$ but is otherwise greater than $J(x)$. The minimizer of $G_k(x)$, shown as a green dot in (b), is taken as x_{k+1}. Plot (c) illustrates the subsequent iteration. In (c) the majorizer coincides with $J(x)$ at $x = x_{k+1}$. The iterates x_k converge to the minimum of $J(x)$. The MM procedure is useful when $J(x)$ is difficult to minimize but each majorizer $G_k(x)$ is easy to minimize.

Figure 1 illustrates the MM procedure with a simple example. For clarity, the figure illustrates the minimization of a function of one variable. However, the MM procedure works in the same way for the minimization of multivariate functions, and it is in the multivariate case where the MM procedure is especially useful for the minimization of multivariate functions. (In the following sections, MM will be used for the minimization of multivariate functions.)

To summarize, the majorization-minimization algorithm for the minimization of a function $J(\mathbf{x})$ is given by the following iteration:

(1) Set $k = 0$. Initialize $\mathbf{x}_0$.
(2) Choose $G_k(\mathbf{x})$ such that
  (a) $G_k(\mathbf{x}) \geq J(\mathbf{x})$ for all $\mathbf{x}$
  (b) $G_k(\mathbf{x}_k) = J(\mathbf{x}_k)$
(3) Set $\mathbf{x}_{k+1}$ as the minimizer of $G_k(\mathbf{x})$.
(4) Set $k = k + 1$ and go to step (2.)

More details about the MM procedure are provided in [8] and the references therein.

3. The Landweber Iteration

Although we are interested in minimizing $J(\mathbf{x})$ in (1), let us consider first the minimization of the simpler objective function

$$\begin{aligned} J(\mathbf{x}) &= \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 && (2) \\ &= (\mathbf{y} - \mathbf{H}\mathbf{x})^{\mathsf{T}}(\mathbf{y} - \mathbf{H}\mathbf{x}) && (3) \\ &= \mathbf{y}^{\mathsf{T}}\mathbf{y} - 2\mathbf{y}^{\mathsf{T}}\mathbf{H}\mathbf{x} + \mathbf{x}^{\mathsf{T}}\mathbf{H}^{\mathsf{T}}\mathbf{H}\mathbf{x}. && (4) \end{aligned}$$

Because $J(\mathbf{x})$ in (2) is differentiable and convex, we can obtain its minimizer by setting the derivative with respect to $\mathbf{x}$ to zero. The derivative of $J(\mathbf{x})$ is given by

$$\frac{\partial}{\partial \mathbf{x}} J(\mathbf{x}) = -2\mathbf{H}^{\mathsf{T}}\mathbf{y} + 2\mathbf{H}^{\mathsf{T}}\mathbf{H}\mathbf{x}.$$

Setting the derivative to zero gives a system of linear equations:

$$\frac{\partial}{\partial \mathbf{x}} J(\mathbf{x}) = \mathbf{0} \quad \Longrightarrow \quad (\mathbf{H}^{\mathsf{T}}\mathbf{H})\,\mathbf{x} = \mathbf{H}^{\mathsf{T}}\mathbf{y}.$$

So the minimizer of $J(\mathbf{x})$ in (2) is given by

$$\mathbf{x} = (\mathbf{H}^{\mathsf{T}}\mathbf{H})^{-1}\mathbf{H}^{\mathsf{T}}\mathbf{y}. \tag{5}$$

Therefore, $J(\mathbf{x})$ in (2) can be minimized by solving a linear system of equations. However, we may not be able to solve these equations easily. For example, if $\mathbf{x}$ is a very long signal, then $\mathbf{H}$ will be very large matrix and solving the system of equations may require too much memory and computation time. Additionally, the matrix $\mathbf{H}^{\mathsf{T}}\mathbf{H}$ might not be invertible, or it may very ill-conditioned.

By using the majorization-minimization (MM) approach to minimize $J(\mathbf{x})$ in (2) we can avoid solving a system of linear equations. As described in Section 2, at each iteration k of the MM approach, we should find a function $G_k(\mathbf{x})$ that coincides with $J(\mathbf{x})$ at $\mathbf{x}_k$ but otherwise upper-bounds $J(\mathbf{x})$. We should choose a majorizer $G_k(\mathbf{x})$ which can be minimized more easily (without having to solve a system of equations).

We will find a function $G_k(\mathbf{x})$ that majorizes $J(\mathbf{x})$ by adding a non-negative function to $J(\mathbf{x})$,

$$G_k(\mathbf{x}) = J(\mathbf{x}) + \text{non-negative function of } \mathbf{x}.$$

In order that $G_k(\mathbf{x})$ coincides with $J(\mathbf{x})$ at $\mathbf{x} = \mathbf{x}_k$, the non-negative function we add to $J(\mathbf{x})$ should be equal to zero at $\mathbf{x}_k$. With this in mind, we choose $G_k(\mathbf{x})$ to be:

$$G_k(\mathbf{x}) = \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 + \underbrace{(\mathbf{x} - \mathbf{x}_k)^{\mathsf{T}}(\alpha\,\mathbf{I} - \mathbf{H}^{\mathsf{T}}\mathbf{H})(\mathbf{x} - \mathbf{x}_k)}_{\text{non-negative}}. \tag{6}$$

The function we have added to $J(\mathbf{x})$ is clearly zero at $\mathbf{x}_k$ so we have $G_k(\mathbf{x}) = J(\mathbf{x}_k)$ as desired.

To ensure the function we have added to $J(\mathbf{x})$ is non-negative for all $\mathbf{x}$, the scalar parameter α must be chosen to be equal to or greater than the maximum eigenvalue of $\mathbf{H}^\mathsf{T}\mathbf{H}$,

$$\alpha \geq \operatorname{maxeig}(\mathbf{H}^\mathsf{T}\mathbf{H}).$$

Then the matrix $\alpha\,\mathbf{I} - \mathbf{H}^\mathsf{T}\mathbf{H}$ is a positive semi-definite matrix, meaning that

$$\mathbf{v}^\mathsf{T}(\alpha\,\mathbf{I} - \mathbf{H}^\mathsf{T}\mathbf{H})\mathbf{v} \geq 0 \quad \forall \mathbf{v}.$$

Now, following the MM procedure, we need to minimize $G_k(\mathbf{x})$ to obtain $\mathbf{x}_{k+1}$. Expanding $G_k(\mathbf{x})$ in (6) gives

$$\begin{aligned} G_k(\mathbf{x}) &= \mathbf{y}^\mathsf{T}\mathbf{y} - 2\mathbf{y}^\mathsf{T}\mathbf{H}\mathbf{x} + \mathbf{x}^\mathsf{T}\mathbf{H}^\mathsf{T}\mathbf{H}\mathbf{x} + (\mathbf{x}-\mathbf{x}_k)^\mathsf{T}(\alpha\,\mathbf{I} - \mathbf{H}^\mathsf{T}\mathbf{H})(\mathbf{x}-\mathbf{x}_k) && (7) \\ &= \mathbf{y}^\mathsf{T}\mathbf{y} + \mathbf{x}_k^\mathsf{T}\,(\alpha\,\mathbf{I} - \mathbf{H}^\mathsf{T}\mathbf{H})\,\mathbf{x}_k - 2\,(\mathbf{y}^\mathsf{T}\mathbf{H} + \mathbf{x}_k^\mathsf{T}(\alpha\,\mathbf{I} - \mathbf{H}^\mathsf{T}\mathbf{H}))\,\mathbf{x} + \alpha\,\mathbf{x}^\mathsf{T}\mathbf{x}. && (8) \end{aligned}$$

Note that the quadratic term in (8) is simply $\alpha\,\mathbf{x}^\mathsf{T}\mathbf{x}$ instead of $\mathbf{x}^\mathsf{T}\mathbf{H}^\mathsf{T}\mathbf{H}\mathbf{x}$ as it was in (4). Therefore we can minimize $G_k(\mathbf{x})$ more easily:

$$\frac{\partial}{\partial \mathbf{x}} G_k(\mathbf{x}) = -2\mathbf{H}^\mathsf{T}\mathbf{y} - 2\,(\alpha\,\mathbf{I} - \mathbf{H}^\mathsf{T}\mathbf{H})\,\mathbf{x}_k + 2\alpha\mathbf{x}$$

$$\frac{\partial}{\partial \mathbf{x}} G_k(\mathbf{x}) = \mathbf{0} \quad \implies \quad \mathbf{x} = \mathbf{x}_k + \frac{1}{\alpha}\mathbf{H}^\mathsf{T}(\mathbf{y} - \mathbf{H}\mathbf{x}_k)$$

Hence, we obtain through the MM procedure, the update equation

$$\boxed{\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{1}{\alpha}\mathbf{H}^\mathsf{T}(\mathbf{y} - \mathbf{H}\mathbf{x}_k)} \qquad (9)$$

which is known as the 'Landweber' iteration. It is guaranteed that $J(\mathbf{x})$ in (2) is decreased with each iteration, due to the properties of the MM procedure. Note that the (9) does not require the solution to a linear system of equations. It only requires multiplying by $\mathbf{H}$ and by $\mathbf{H}^\mathsf{T}$.

To emphasize the functional dependence of $G_k(\mathbf{x})$ on $\mathbf{x}$ we can write $G_k(\mathbf{x})$ in (8) as

$$G_k(\mathbf{x}) = \alpha\,(-2\mathbf{b}^\mathsf{T}\mathbf{x} + \mathbf{x}^\mathsf{T}\mathbf{x}) + c \qquad (10)$$

where

$$\begin{aligned} \mathbf{b} &= \frac{1}{\alpha}\left[\mathbf{H}^\mathsf{T}\mathbf{y} + (\alpha\,\mathbf{I} - \mathbf{H}^\mathsf{T}\mathbf{H})\,\mathbf{x}_k\right] && (11) \\ &= \mathbf{x}_k + \frac{1}{\alpha}\mathbf{H}^\mathsf{T}(\mathbf{y} - \mathbf{H}\mathbf{x}_k) && (12) \end{aligned}$$

and where c consists of the first two terms of (8) which do not depend on $\mathbf{x}$. Note that for any vectors $\mathbf{b}$ and $\mathbf{x}$,

$$\mathbf{b}^\mathsf{T}\mathbf{b} - 2\mathbf{b}^\mathsf{T}\mathbf{x} + \mathbf{x}^\mathsf{T}\mathbf{x} = \|\mathbf{b} - \mathbf{x}\|_2^2,$$

hence from (10) we can write $G_k(\mathbf{x})$ as

$$G_k(\mathbf{x}) = \alpha\|\mathbf{b} - \mathbf{x}\|_2^2 - \alpha\mathbf{b}^\mathsf{T}\mathbf{b} + c.$$

That is, using (12), the majorizer $G_k(\mathbf{x})$ can be written as

$$G_k(\mathbf{x}) = \alpha\left\|\mathbf{x}_k + \frac{1}{\alpha}\mathbf{H}^\mathsf{T}(\mathbf{y} - \mathbf{H}\mathbf{x}_k) - \mathbf{x}\right\|_2^2 + K \qquad (13)$$

where K does not depend on $\mathbf{x}$. This shows that $G_k(\mathbf{x})$ is shaped like a bowl with circular level sets.

4. Soft-Thresholding

As in Section 3, let us consider the minimization of an objective function simpler than (1). Namely, let us consider the minimization of the function

$$J(\mathbf{x}) = \|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda \|\mathbf{x}\|_1. \tag{14}$$

This is a special simple case of (1) with $\mathbf{H} = \mathbf{I}$.

The function $J(\mathbf{x})$ in (14) is convex, but not differentiable (because $\|\mathbf{x}\|_1$ is not differentiable). Nevertheless, the minimizer of this $J(\mathbf{x})$ is given by a simple formula.

Expanding $J(\mathbf{x})$ in (14) gives

$$J(\mathbf{x}) = (y_1 - x_1)^2 + \lambda |x_1| + (y_2 - x_2)^2 + \lambda |x_2| + \cdots + (y_N - x_N)^2 + \lambda |x_N|.$$

Note that the variables, x_i, are *uncoupled.* That is, the function $J(\mathbf{x})$ can be minimized by minimizing each term $(y_i - x_i)^2 + \lambda |x_i|$ individually to get x_i, for $1 \le i \le N$. Therefore we need only consider the scalar minimization of the function

$$f(x) = (y - x)^2 + \lambda |x|. \tag{15}$$

Taking the derivative,

$$f'(x) = -2(y - x) + \lambda \operatorname{sign}(x),$$

setting $f'(x) = 0$ gives

$$y = x + \frac{\lambda}{2} \operatorname{sign}(x).$$

Solving for x gives the graph shown in Fig. 2 with threshold $\lambda/2$. That is, the minimizer of $f(x)$ is obtained by applying the soft-threshold rule to y with threshold $\lambda/2$.

The soft-threshold rule is the the non-linear function defined as

$$\operatorname{soft}(x, T) := \begin{cases} x + T & x \le -T \\ 0 & |x| \le T \\ x - T & x \ge T \end{cases}$$

or more compactly, as

$$\operatorname{soft}(x, T) := \operatorname{sign}(x) \max(0, |x| - T). \tag{16}$$

The soft-threshold rule is illustrated in Fig. 2. In terms of the soft-threshold rule, the minimization of (15) is given by

$$x = \operatorname{soft}(y,\ \lambda/2).$$

Because the variables in the function $J(\mathbf{x})$ in (14) are uncoupled and the solution is obtained by minimizing with respect to each x_i individually, the minimizer of $J(\mathbf{x})$ is obtained by applying the soft-thresholding rule to each element of $\mathbf{x}$:

$$\boxed{\mathbf{x} = \operatorname{soft}\left(\mathbf{y},\ \lambda/2\right).} \tag{17}$$

The minimization of (14) does not require an iterative algorithm. It is minimized simply by soft-thresholding each element of $\mathbf{y}$ with threshold $\lambda/2$.

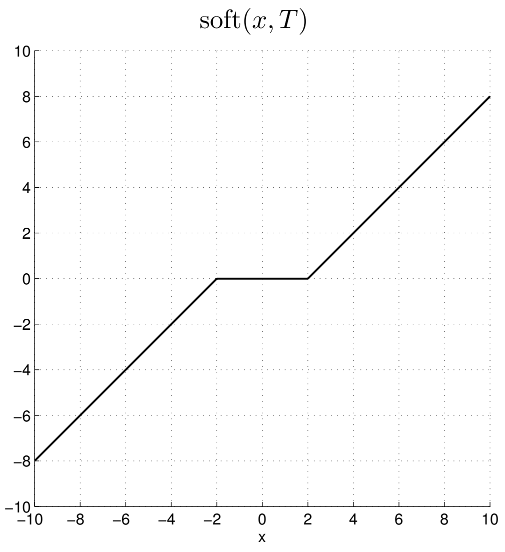

FIGURE 2. The soft-threshold rule (16) with threshold $T = 2$.

5. Iterated Soft-Thresholding Algorithm (ISTA)

In this section, we consider the minimization of the objective function $J(\mathbf{x})$ in (1) which we repeat here,

$$J(\mathbf{x}) = \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_1. \tag{18}$$

This $J(\mathbf{x})$ can be minimized by combining the results of Sections 3 and 4.

Applying the MM approach, we wish to find a majorizer of $J(\mathbf{x})$ which coincides with $J(\mathbf{x})$ at $\mathbf{x}_k$ and which is easily minimized. We can add to $J(\mathbf{x})$ the same non-negative function as in (6) in Section 3,

$$G_k(\mathbf{x}) = J(\mathbf{x}) + (\mathbf{x} - \mathbf{x}_k)^\mathsf{T}(\alpha\,\mathbf{I} - \mathbf{H}^\mathsf{T}\mathbf{H})(\mathbf{x} - \mathbf{x}_k).$$

By design, $G_k(\mathbf{x})$ coincides with $J(\mathbf{x})$ at $\mathbf{x}_k$. We need to minimize $G_k(\mathbf{x})$ to get $\mathbf{x}_{k+1}$. With (18), we obtain

$$G_k(\mathbf{x}) = \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 + (\mathbf{x} - \mathbf{x}_k)^\mathsf{T}(\alpha\,\mathbf{I} - \mathbf{H}^\mathsf{T}\mathbf{H})(\mathbf{x} - \mathbf{x}_k) + \lambda\|\mathbf{x}\|_1$$

which is the same as (6) except here we have the additional term $\lambda\|\mathbf{x}\|_1$. Using (13), we can write $G_k(\mathbf{x})$ as

$$G_k(\mathbf{x}) = \alpha\left\|\mathbf{x}_k + \frac{1}{\alpha}\mathbf{H}^\mathsf{T}(\mathbf{y} - \mathbf{H}\mathbf{x}_k) - \mathbf{x}\right\|_2^2 + \lambda\|\mathbf{x}\|_1 + K$$

where K is a constant with respect to $\mathbf{x}$. Minimizing $G_k(\mathbf{x})$ is equivalent to minimizing $(1/\alpha)\,G_k(\mathbf{x})$, so $\mathbf{x}_{k+1}$ is obtained by minimizing

$$\left\|\mathbf{x}_k + \frac{1}{\alpha}\mathbf{H}^\mathsf{T}(\mathbf{y} - \mathbf{H}\mathbf{x}_k) - \mathbf{x}\right\|_2^2 + \frac{\lambda}{\alpha}\|\mathbf{x}\|_1. \tag{19}$$

We omit the additive constant term because the vector $\mathbf{x}$ minimizing $G_k(\mathbf{x})$ is not influenced by it.

Note that (19) has exactly the same form as (14) which is minimized by the formula (17). Therefore, minimizing (19) is achieved by the following soft-thresholding equation:

$$\boxed{\mathbf{x}_{k+1} = \text{soft}\left(\mathbf{x}_k + \frac{1}{\alpha}\mathbf{H}^\mathsf{T}(\mathbf{y} - \mathbf{H}\mathbf{x}_k),\ \frac{\lambda}{2\alpha}\right)} \tag{20}$$

where $\alpha \geq \text{maxeig}(\mathbf{H}^\mathsf{T}\mathbf{H})$. This gives the iterated soft-thresholding algorithm (ISTA).

The MATLAB program `ista` in Listing 1 implements the iterated soft-thresholding algorithm.

```
function [x,J] = ista(y,H,lambda,alpha,Nit)
% [x, J] = ista(y, H, lambda, alpha, Nit)
% L1-regularized signal restoration using the iterated
% soft-thresholding algorithm (ISTA)
% Minimizes J(x) = norm2(y-H*x)^2 + lambda*norm1(x)
%  INPUT
%   y - observed signal
%   H - matrix or operator
%   lambda - regularization parameter
%   alpha - need alpha >= max(eig(H'*H))
%   Nit - number of iterations
% OUTPUT
%   x - result of deconvolution
%   J - objective function

J = zeros(1, Nit);       % Objective function
x = 0*H'*y;              % Initialize x
T = lambda/(2*alpha);
for k = 1:Nit
    Hx = H*x;
    J(k) = sum(abs(Hx(:)-y(:)).^2) + lambda*sum(abs(x(:)));
    x = soft(x + (H'*(y - Hx))/alpha, T);
end
```

Listing 1: A MATLAB program implementing the iterated soft-thresholding algorithm (ISTA). The main update equation is (20).

6. Example

Figures 3 and 4 illustrate an example of the ℓ_1-norm regularized signal restoration procedure.

A clean sparse signal, $x(n)$, of 100 samples in duration is illustrated in Fig. 3. The observed signal is obtained by convolving $x(n)$ by the impulse response

$$h = [1,\ 2,\ 3,\ 4,\ 3,\ 2,\ 1]/16$$

and adding white Gaussian noise with standard deviation 0.05.

The result of 500 iterations of ISTA is illustrated in Fig. 4. The decay of the objective function $J(\mathbf{x})$ is also illustrated. The restored signal is not exactly the same as the original signal, however, it is quite close.

To implement this example in MATLAB, we used the following commands:

```
h = [1 2 3 4 3 2 1]/16;
N = 100;
H = convmtx(h',N);
lambda = 0.1;
alpha = 1;
Nit = 500;
[x, J] = ista(y, H, lambda, alpha, Nit);
```

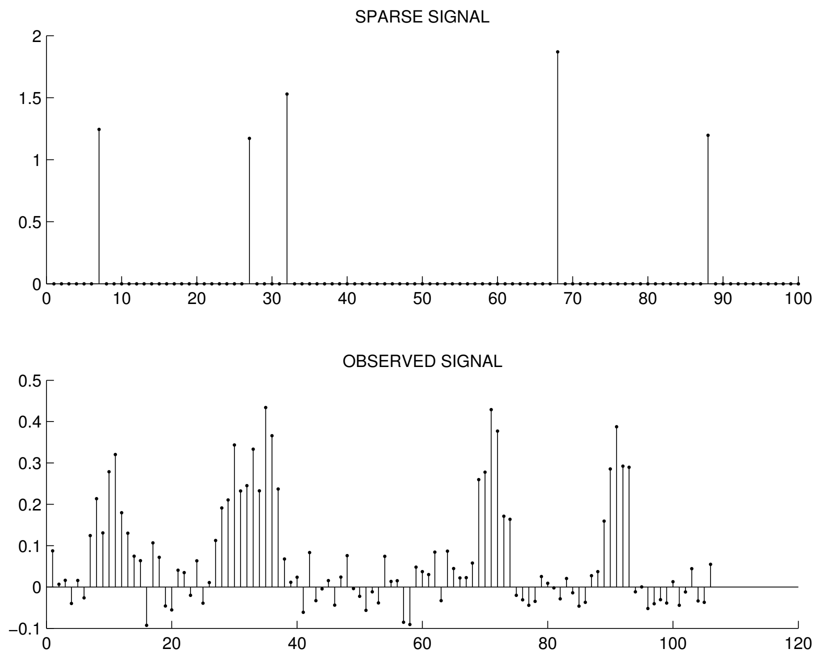

FIGURE 3. Example 1: The sparse signal x and the observed signal $y = h * x + \text{noise}$.

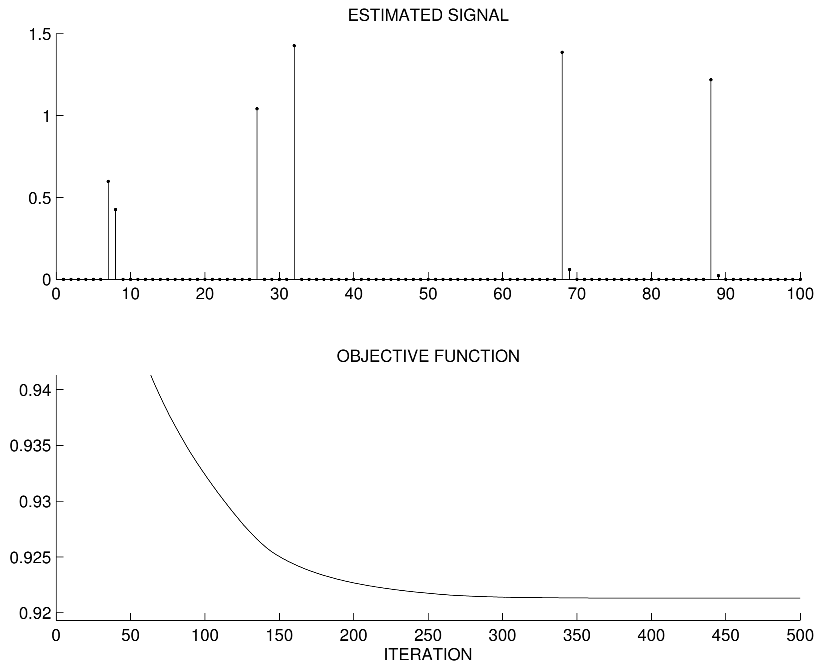

FIGURE 4. Example 1: The estimated signal obtained by minimizing the objective function (1). The objective function $J(\mathbf{x})$ versus iteration number.

```
function f = convt(h,g);
% f = convt(h,g);
% Transpose convolution: f = H' g
Nh = length(h);
Ng = length(g);
f = conv(h(Nh:-1:1), g);
f = f(Nh:Ng);
```

Listing 2: A MATLAB program for transpose convolution as in (22).

7. Convolution and its transpose

Convolution can be represented by matrix-vector multiplication:

$$\mathbf{H}\mathbf{x} = \begin{bmatrix} h_0 & & & & \\ h_1 & h_0 & & & \\ h_2 & h_1 & h_0 & & \\ & h_2 & h_1 & h_0 & \\ & & h_2 & h_1 & h_0 \\ & & & h_2 & h_1 \\ & & & & h_2 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}. \tag{21}$$

In MATLAB, we can use `conv(h,x)` to implement (21). We do not need to perform matrix-vector multiplication. Using `conv` instead of the matrix $\mathbf{H}$ is preferable in a computer program so as to avoid storing the matrix (the matrix $\mathbf{H}$ will be very large when the signal $\mathbf{x}$ is long).

The matrix $\mathbf{H}^{\mathsf{T}}$ is similar but not identical:

$$\mathbf{H}^{\mathsf{T}}\mathbf{g} = \begin{bmatrix} h_0 & h_1 & h_2 & & & & \\ & h_0 & h_1 & h_2 & & & \\ & & h_0 & h_1 & h_2 & & \\ & & & h_0 & h_1 & h_2 & \\ & & & & h_0 & h_1 & h_2 \end{bmatrix} \begin{bmatrix} g_0 \\ g_1 \\ g_2 \\ g_3 \\ g_4 \\ g_5 \\ g_6 \end{bmatrix}. \tag{22}$$

This is a truncation of convolution with a reversed version of $\mathbf{h}$:

$$\left[\begin{array}{ccccccc} h_2 & & & & & & \\ h_1 & h_2 & & & & & \\ \hline h_0 & h_1 & h_2 & & & & \\ & h_0 & h_1 & h_2 & & & \\ & & h_0 & h_1 & h_2 & & \\ & & & h_0 & h_1 & h_2 & \\ & & & & h_0 & h_1 & h_2 \\ \hline & & & & & h_0 & h_1 \\ & & & & & & h_0 \end{array}\right] \begin{bmatrix} g_0 \\ g_1 \\ g_2 \\ g_3 \\ g_4 \\ g_5 \\ g_6 \end{bmatrix}. \tag{23}$$

So we can implement $\mathbf{f} = \mathbf{H}^{\mathsf{T}}\,\mathbf{g}$ in MATLAB using the program `convt` in Listing 2.

The MATLAB program `ista_fns` in Listing 3 uses function handles for $\mathbf{H}$ and $\mathbf{H}^{\mathsf{T}}$ instead of matrices, so it is not necessary to create the matrix $\mathbf{H}$ in MATLAB. Instead, it is necessary to supply a function for

```
function [x,J] = ista_fns(y,H,Ht,lambda,alpha,Nit)
% [x, J] = ista_fns(y, H, lambda, alpha, Nit)
% L1-regularized signal restoration using the iterated
% soft-thresholding algorithm (ISTA)
% Minimizes J(x) = norm2(y-H*x)^2 + lambda*norm1(x)
%  INPUT
%   y - observed signal
%   H - function handle
%   Ht - function handle for H'
%   lambda - regularization parameter
%   alpha - need alpha >= max(eig(H'*H))
%   Nit - number of iterations
% OUTPUT
%   x - result of deconvolution
%   J - objective function

J = zeros(1, Nit);       % Objective function
x = 0*Ht(y);             % Initialize x
T = lambda/(2*alpha);
for k = 1:Nit
    Hx = H(x);
    J(k) = sum(abs(Hx(:)-y(:)).^2) + lambda*sum(abs(x(:)));
    x = soft(x + (Ht(y - Hx))/alpha, T);
end
```

Listing 3: A MATLAB program implementing the iterated soft-thresholding algorithm (ISTA). This program uses functions handles for $\mathbf{H}$ and $\mathbf{H}^\mathsf{T}$ so as to avoid the use of large matrices.

each of $\mathbf{H}$ and $\mathbf{H}^\mathsf{T}$. For example, in the Example where $\mathbf{H}$ represents convolution, we can specify function handles as follows:

```
h = [1 2 3 4 3 2 1]/16;
H = @(x) conv(h,x);
Ht = @(y) convt(h,y);
lambda = 0.1;
alpha = 1;
Nit = 500;
[x, J] = ista_fns(y, H, Ht, lambda, alpha, Nit);
```

The result is exactly the same as using the `ista` program.

8. Conclusion

These notes describe ISTA (iterated soft-thresholding algorithm) for solving ℓ_1-norm regularized inverse problems. ISTA has linear convergence. There are several more recently developed algorithms that have faster convergence properties than ISTA. For example, FISTA (Fast-ISTA), described in [1] has quadratic convergence.

In place of the ℓ_1-norm, an ℓ_p-norm with $p < 1$ can be used to more strongly promote sparsity in the solution. However, in this case, the cost function $J(\mathbf{x})$ will not be convex and the optimization problem is therefore more difficult in general.

Appendix A. Vector Derivatives

Suppose $\mathbf{x}$ is an N-point vector,

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix},$$

then the derivative of a function $f(\mathbf{x})$ with respect to $\mathbf{x}$ is the vector of derivatives,

$$\frac{\partial}{\partial \mathbf{x}} f(\mathbf{x}) = \begin{bmatrix} \dfrac{\partial f(\mathbf{x})}{\partial x_1} \\ \dfrac{\partial f(\mathbf{x})}{\partial x_2} \\ \vdots \\ \dfrac{\partial f(\mathbf{x})}{\partial x_N} \end{bmatrix}.$$

By direct calculation, we have

$$\frac{\partial}{\partial \mathbf{x}} \mathbf{b}^\mathsf{T} \mathbf{x} = \mathbf{b}.$$

Suppose that $\mathbf{A}$ is a symmetric real matrix, $\mathbf{A}^\mathsf{T} = \mathbf{A}$. Then, by direct calculation, we also have

$$\frac{\partial}{\partial \mathbf{x}} \mathbf{x}^\mathsf{T} \mathbf{A} \mathbf{x} = 2\mathbf{A}\mathbf{x}$$

and

$$\frac{\partial}{\partial \mathbf{x}} (\mathbf{y} - \mathbf{x})^\mathsf{T} \mathbf{A} (\mathbf{y} - \mathbf{x}) = 2\mathbf{A}(\mathbf{x} - \mathbf{y}).$$

Appendix B. ℓ_2 Regularization

Another standard linear inverse problem formulation uses ℓ_2-norm regularization:

$$J(\mathbf{x}) = \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 + \lambda \|\mathbf{x}\|_2^2 \tag{24}$$

where $\lambda > 0$. Taking the derivative gives

$$\frac{\partial}{\partial \mathbf{x}} J(\mathbf{x}) = 2\mathbf{H}^\mathsf{T}(\mathbf{H}\mathbf{x} - \mathbf{y}) + 2\lambda\mathbf{x}.$$

Setting the derivative to zero,

$$\frac{\partial}{\partial \mathbf{x}} J(\mathbf{x}) = \mathbf{0} \quad \Longrightarrow \quad \mathbf{H}^\mathsf{T}\mathbf{H}\mathbf{x} + \lambda\mathbf{x} = \mathbf{H}^\mathsf{T}\mathbf{y} \quad \Longrightarrow \quad (\mathbf{H}^\mathsf{T}\mathbf{H} + \lambda\mathbf{I})\,\mathbf{x} = \mathbf{H}^\mathsf{T}\mathbf{y}$$

So the solution is given by

$$\boxed{\mathbf{x} = (\mathbf{H}^\mathsf{T}\mathbf{H} + \lambda\mathbf{I})^{-1}\mathbf{H}^\mathsf{T}\mathbf{y}} \tag{25}$$

This is referred to as 'diagonal loading' because a constant, λ, is added to the diagonal elements of $\mathbf{H}^\mathsf{T}\mathbf{H}$. The approach also avoids the problem of rank deficiency because $\mathbf{H}^\mathsf{T}\mathbf{H} + \lambda\mathbf{I}$ is invertible even if $\mathbf{H}^\mathsf{T}\mathbf{H}$ is not.

Equation (25) is to be compared with (5): if $\lambda = 0$, then (25) gives (5).

B.1. **Landweber iteration for ℓ_2 regularized inverse problem.** Make a majorizer:

$$G_k(\mathbf{x}) = \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 + \lambda \|\mathbf{x}\|_2^2 + \underbrace{(\mathbf{x} - \mathbf{x}_k)^\mathsf{T}(\alpha\,\mathbf{I} - \mathbf{H}^\mathsf{T}\mathbf{H})(\mathbf{x} - \mathbf{x}_k)}_{\text{non-negative}} \tag{26}$$

Expanding:

$$\begin{aligned} G_k(\mathbf{x}) &= \mathbf{x}^\mathsf{T}\mathbf{H}^\mathsf{T}\mathbf{H}\mathbf{x} - 2\mathbf{y}^\mathsf{T}\mathbf{H}\mathbf{x} + \mathbf{y}^\mathsf{T}\mathbf{y} + \lambda\mathbf{x}^\mathsf{T}\mathbf{x} + (\mathbf{x} - \mathbf{x}_k)^\mathsf{T}(\alpha\,\mathbf{I} - \mathbf{H}^\mathsf{T}\mathbf{H})(\mathbf{x} - \mathbf{x}_k) && (27) \\ &= (\alpha + \lambda)\,\mathbf{x}^\mathsf{T}\mathbf{x} - 2(\mathbf{y}^\mathsf{T}\mathbf{H} + \mathbf{x}_k^\mathsf{T}(\alpha\,\mathbf{I} - \mathbf{H}^\mathsf{T}\mathbf{H}))\mathbf{x} + \mathbf{y}^\mathsf{T}\mathbf{y} + \mathbf{x}_k^\mathsf{T}(\alpha\,\mathbf{I} - \mathbf{H}^\mathsf{T}\mathbf{H})\mathbf{x}_k && (28) \end{aligned}$$

The leading term of $G_k(\mathbf{x})$ is $(\alpha + \lambda)\mathbf{x}^\mathsf{T}\mathbf{x}$ so we can minimize $G_k(\mathbf{x})$ easily:

$$\frac{\partial}{\partial \mathbf{x}} G_k(\mathbf{x}) = 2(\alpha + \lambda)\mathbf{x} - 2\mathbf{H}^\mathsf{T}\mathbf{y} - 2(\alpha\,\mathbf{I} - \mathbf{H}^\mathsf{T}\mathbf{H})\mathbf{x}_k$$

$$\frac{\partial}{\partial \mathbf{x}} G_k(\mathbf{x}) = \mathbf{0} \quad\implies\quad \mathbf{x} = \frac{\alpha}{\alpha + \lambda}\mathbf{x}_k + \frac{1}{\alpha + \lambda}\mathbf{H}^\mathsf{T}(\mathbf{y} - \mathbf{H}\mathbf{x}_k)$$

so we get the iteration:

$$\boxed{\mathbf{x}_{k+1} = \frac{\alpha}{\alpha + \lambda}\mathbf{x}_k + \frac{1}{\alpha + \lambda}\mathbf{H}^\mathsf{T}(\mathbf{y} - \mathbf{H}\mathbf{x}_k)} \tag{29}$$

Note that if $\lambda = 0$ then (29) becomes (9).

References

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

[2] J. M. Bioucas-Dias and M. A. T. Figueiredo. A new TwIST: Two-step iterative shrinkage/thresholding algorithms for image restoration. *IEEE Trans. Image Process.*, 16(12):2992–3004, December 2007.

[3] P. L. Combettes and J.-C. Pesquet. Proximal thresholding algorithm for minimization over orthonormal bases. *SIAM J. on Optimization*, 18(4):1351–1376, 2007.

[4] 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.

[5] I. Daubechies, M. Fornasier, and I. Loris. Accelerated projected gradient method for linear inverse problems with sparsity constraints. *Journal of Fourier Analysis and Applications*, 14:764–792, December 2008.

[6] I. Daubechies, G. Teschke, and L. Vese. On some iterative concepts for image restoration. In P. W. Hawkes, editor, *Advances in Imaging and Electron Physics*, volume 150, pages 2–51. Elsevier, 2008.

[7] M. Elad, B. Matalon, and M. Zibulevsky. Coordinate and subspace optimization methods for linear least squares with non-quadratic regularization. *J. of Appl. and Comp. Harm. Analysis*, 23:346–367, 2007.

[8] M. Figueiredo, J. Bioucas-Dias, and R. Nowak. Majorization-minimization algorithms for wavelet-based image restoration. *IEEE Trans. Image Process.*, 16(12):2980–2991, December 2007.

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

[10] M. A. T. Figueiredo, R. D. Nowak, and S. J. Wright. Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE J. Sel. Top. Signal Process.*, 1(4):586–598, December 2007.

[11] T. Goldstein and S. Osher. The split Bregman method for L1-regularized problems. *SIAM J. Imag. Sci.*, 2(2):323–343, 2009.

[12] E. T. Hale, W. Yin, and Y. Zhang. Fixed-point continuation for ℓ_1-minimization: Methodology and convergence. *SIAM J. on Optimization*, 19(3):1107–1130, 2008.

[13] Y. Wang, J. Yang, W. Yin, and Y. Zhang. A new alternating minimization algorithm for total variation image reconstruction. *SIAM J. on Imaging Sciences*, 1(3):248–272, 2008.

[14] S. J. Wright, R. D. Nowak, and M. A. T. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Trans. Signal Process.*, 57(7):2479–2493, July 2009.