Sparsity-Assisted Signal Smoothing

Ivan W. Selesnick*

Electrical and Computer Engineering
NYU Polytechnic School of Engineering
Brooklyn, NY, 11201
selesi@nyu.edu

Summary. This chapter describes a method for one-dimensional signal denoising that simultaneously utilizes both sparse optimization principles and conventional linear time-invariant (LTI) filtering. The method, called 'sparsity-assisted signal smoothing' (SASS), is based on modeling a signal as the sum of a low-pass component and a piecewise smooth component. The problem is formulated as a sparse-regularized linear inverse problem. We provide simple direct methods to set the regularization and non-convexity parameters, the later if a non-convex penalty is utilized. We derive an iterative optimization algorithm that harnesses the computational efficiency of fast solvers for banded systems. The SASS approach performs a type of wavelet denoising, but does so through sparse optimization rather than through wavelet transforms. The approach is relatively free of the pseudo-Gibbs phenomena that tends to arise in wavelet denoising.

Key words: filtering, total variation denoising, wavelet denoising, convex optimization, sparse optimization

1 Introduction

This chapter develops a method for noise reduction, called 'sparsity-assisted signal smoothing' (SASS). The proposed SASS approach models the unknown signal of interest, $x(t)$, as the sum,

$$x(t) = f(t) + g(t), \tag{1}$$

where $f(t)$ is a low-pass signal, and $g(t)$ has a sparse order-K derivative. We consider the problem of estimating $x(t)$ from noisy data, $y(t) = x(t) + w(t)$, where $w(t)$ is white Gaussian noise. The SASS denoising approach combines the principles of sparse optimization with conventional linear time-invariant (LTI) filtering.[2]

* This research was supported by the NSF under grant CCF-1018020.

[2] Software is available at http://eeweb.poly.edu/iselesni/sass/

The SASS method involves low-pass filtering and the minimization of a non-differentiable objective function that promotes sparsity of the order-K derivative of $g(t)$. We formulate the SASS approach as a sparse-regularized linear inverse problem, which after a change of variables, is shown to be a sparse deconvolution problem. Both convex and non-convex regularizations are considered. In both cases, we provide a simple, direct method to set the regularization parameter. In the non-convex case, we also provide a method to set the parameter that controls the degree of non-convexity.

A computationally efficient iterative optimization algorithm is developed for the SASS approach. The SASS approach is intentionally constructed using banded matrices exclusively, so fast solvers for banded systems can be used for its implementation. The optimization algorithm calls for no parameters (step sizes, etc.). In addition, we describe a method for dealing with the zero-locking issue, which can arise in the non-convex case. The method detects and corrects zero-locking, when it occurs.

The SASS approach builds upon and extends the practice and capabilities of conventional low-pass filtering [44]. The first step of the method involves the specification of a low-pass filter, the cut-off frequency of which can be set as usual, according to knowledge of the frequency spectrum of the signals in question. In one limiting case ($\lambda \to \infty$), SASS amounts to low-pass filtering. In another limiting case ($\lambda \to 0$), SASS performs no filtering and the output of the method is the noisy input data. For practical values of λ, the SASS approach can be understood as an enhancement of the low-pass filter, as will be illustrated.

1.1 Related work

Several recent works have utilized an approach wherein a signal is explicitly expressed as the sum of a low-pass signal and a piecewise constant (i.e., sparse-derivative) signal [30, 51, 53]. In each approach, an inverse problem is formulated where total variation regularization [49] is used to estimate the piecewise constant signal component. These methods differ in the way the low-pass signal component is modeled and estimated. The low-pass component is modeled as locally polynomial in [51], while Tikhonov regularization is used in [30], and conventional low-pass filtering in [53].

The signal model used in these works (low-pass plus piecewise constant) is well suited for applications where additive step discontinuities are observed in the presence of a relatively slow varying signal. For example, the methods described in [51] and [53] are demonstrated respectively on data produced by a nano-particle biosensor [16] and a near infrared spectroscopic (NIRS) imaging device [1]. However, this model is limited because many natural (e.g., physiological) signals do not exhibit additive step discontinuities. Instead, they are more accurately modeled as having discontinuities in a higher order derivative.

The problem addressed in this chapter is an extension of one of the problems addressed in [53]. Specifically, SASS extends the 'LPF/TVD' problem [53] from $K = 1$ to $K > 1$, where K is the order of the sparse derivative. The LPF/TVD algorithm in [53] is not effective for the case $K > 1$, because it estimates the sparse-derivative component by the integration of a sparse signal. Using this approach for $K > 1$ leads to K-order integration which is very unstable; the obtained sparse order-K derivative component will inevitably be unbounded. The SASS approach in this chapter circumvents this problem by estimating the sum, $x = f + g$ in (1), without estimating f and g individually.

The SASS approach can also be viewed as an extension of one-dimensional total variation (TV) denoising [8,9,49]. TV denoising is based on the assumption that the derivative of $x(t)$ is sparse, i.e., that $x(t)$ is approximately piecewise constant. Total variation denoising is notable due to its ability to preserve discontinuities and the absence of pseudo-Gibbs phenomena. However, TV denoising is afflicted by staircase artifacts and performs poorly for more general signals. Hence numerous generalizations for TV have been proposed to make TV denoising more widely effective, e.g., higher-order TV, directional TV, etc. [2, 32, 34, 39]. The proposed SASS approach also uses higher-order TV, but in contrast to these methods, SASS incorporates a low-pass filter (LPF). The incorporation of the low-pass filter enhances both the prospective sparsity of the order-K derivative and the flexibility of high-order TV regularization. In effect, the LPF lifts some of the burden off the high-order total variation regularization.

Wavelet-based signal denoising is also suitably applied to signals of the form considered in this work. Several wavelet-domain algorithms have been developed specifically to account for the presence of singularities (i.e., discontinuities in the signal or its derivatives). In order to suppress spurious oscillations (the pseudo-Gibbs phenomenon) around singularities, which arise due to the modification of wavelet coefficients, these algorithms generally impose some model or constraints in the wavelet domain. Examples of such approaches include: wavelet hidden Markov tree (HMT) [15], singularity detection [31, 33], wavelet footprints [20, 56], total variation regularization [21, 22], and singularity approximation [4, 5]. The proposed SASS approach is similar to these techniques in that it accounts for singularities in the signal; however, it does so through sparse optimization instead of wavelet transforms.

2 Preliminaries

2.1 Notation

We represent finite-length discrete-time signals as vectors and denote them in lower-case, e.g., we represent the N-point signal $\mathbf{x} \in \mathbb{R}^N$ as

$$\mathbf{x} = [x(0), \ldots, x(N-1)]^{\mathsf{T}} \tag{2}$$

where $[\,\cdot\,]^{\mathsf{T}}$ denotes the transpose. Matrices are denoted in upper case, e.g., $\mathbf{H} \in \mathbb{R}^{L \times N}$.

The notations $\|\mathbf{v}\|_1$ and $\|\mathbf{v}\|_2$ denote the ℓ_1 and ℓ_2 norms of the vector $\mathbf{v}$ respectively; i.e.,

$$\|\mathbf{v}\|_1 = \sum_n |v(n)|, \qquad \|\mathbf{v}\|_2^2 = \sum_n |v(n)|^2. \tag{3}$$

We denote the order-K difference matrix by $\mathbf{D}$. For example, the second-order difference matrix ($K = 2$) is given by,

$$\mathbf{D} = \begin{bmatrix} 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & \ddots & & \\ & & & 1 & -2 & 1 \end{bmatrix} \tag{4}$$

where $\mathbf{D}$ is of size $(N-2) \times N$. The second-order difference of an N-point signal $\mathbf{x}$ is then given by $\mathbf{Dx}$.

In general, $\mathbf{D}$ is a Toeplitz matrix of size $(N-K) \times N$. For $K = 1$, the first row of $\mathbf{D}$ is $[-1,\ 1]$. For $K = 3$, the first row is $[-1,\ 3,\ -3,\ 1]$. In general, the first row of $\mathbf{D}$ consists of the coefficients of $(1-z)^K$. The order-K difference, $\mathbf{D} : \mathbb{R}^N \to \mathbb{R}^{N-K}$, is precisely defined by $\mathbf{y} = \mathbf{Dx}$, where

$$y(n) = \sum_{k=0}^{K} (-1)^k \binom{K}{k} x(n + K - k),$$

for $0 \leqslant n \leqslant N - K - 1$. Note that $\mathbf{D}$ annihilates polynomial of degree $K - 1$.

2.2 Filters

LTI filters are usually implemented via recursive difference equations [44]; however, in this work, we use banded Toeplitz matrices. We do so, because this facilitates incorporating LTI filtering into the sparse optimization framework. It also provides a simple mechanism to perform zero-phase non-causal recursive filtering of finite-length signals. For example, the difference equation

$$a_1 y(n+1) + a_0 y(n) + a_1 y(n-1) = b_1 x(n+1) + b_0 x(n) + b_1 x(n-1) \tag{5}$$

can be written and implemented as

$$\mathbf{y} = \mathbf{A}^{-1}\mathbf{B}\mathbf{x} \tag{6}$$

where $\mathbf{A}$ is a square banded Toeplitz matrix of the form

$$\mathbf{A} = \begin{bmatrix} a_0 & a_1 & & & \\ a_1 & a_0 & a_1 & & \\ & \ddots & & & \\ & & a_1 & a_0 & a_1 \\ & & & a_1 & a_0 \end{bmatrix} \tag{7}$$

and $\mathbf{B}$ is similarly a banded Toeplitz matrix (not necessarily square), e.g.,

$$\mathbf{B} = \begin{bmatrix} b_1 & b_0 & b_1 & & \\ & b_1 & b_0 & b_1 & \\ & & \ddots & & \\ & & & b_1 & b_0 & b_1 \end{bmatrix}. \tag{8}$$

We define the filter matrix, $\mathbf{H}$, as

$$\mathbf{H} = \mathbf{A}^{-1}\mathbf{B} \tag{9}$$

which can be implemented using fast solvers for banded systems [46, Sect 2.4]. Note that the order-K difference matrix, $\mathbf{D}$, represents the filter with transfer function $D(z) = (1 - z^{-1})^K$.

In this work, we use the high-pass filter

$$H(z) = \frac{B(z)}{A(z)} = \frac{(-z + 2 - z^{-1})^d}{(-z + 2 - z^{-1})^d + \alpha\,(z + 2 + z^{-1})^d} \tag{10}$$

where $\alpha > 0$ is used to set the cut-off frequency. This is a zero-phase Butterworth filter of order $2d$. The filter has a zero at $z = 1$ of order $2d$. We implement this filter as $\mathbf{H} = \mathbf{A}^{-1}\mathbf{B}$.

Note that, if $K \leqslant 2d$, then the numerator, $\mathbf{B}$, of this high-pass filter satisfies

$$\mathbf{B} = \mathbf{B}_1\mathbf{D} \tag{11}$$

where $\mathbf{B}_1$ is banded and $\mathbf{D}$ is the order-K difference matrix. In terms of transfer functions, (11) means that $B(z) = B_1(z)D(z)$ and hence, $B_1(z)$ has a zero of multiplicity $2d - K$ at $z = 1$.

In Sec. 3.2, the filter $\mathbf{A}^{-1}\mathbf{B}_1$ will arise, which we denote as $\mathbf{H}_1$. The transfer function of this filter, $H_1(z)$, is the same as $H(z)$ in (10), but with K fewer zeros at $z = 1$. Furthers details about the filters are given in [53].

The implementation of a filter as $\mathbf{y} = \mathbf{A}^{-1}\mathbf{B}\mathbf{x}$ does tend to produce transients at the beginning and end of the signal. In practice, we alleviate these transients by polynomial smoothing of the first and last few signal values prior to filtering, as discussed in Section 4.4.

3 Problem Formulation

In the sparsity-assisted signal smoothing (SASS) approach, it is assumed that the N-point discrete-time data, y, is of the form

$$\mathbf{y} = \mathbf{f} + \mathbf{g} + \mathbf{w}, \quad \mathbf{y}, \mathbf{f}, \mathbf{g}, \mathbf{w} \in \mathbb{R}^N \tag{12}$$

where f is a low-pass signal, g is a signal with (approximately) sparse order-K derivative, and w is white Gaussian noise. The assumption that the order-K

derivative of g is sparse implies that g is (approximately) piecewise polynomial, where each polynomial segment is of degree $K-1$. However, the approach described here does not explicitly parameterize the signal in terms of its polynomial coefficients, nor does it explicitly segment the signal into polynomial segments (cf., splines).

We further assume that if the signal component g were absent in (12), then a low-pass filter can be used to satisfactorily estimate f from the data, y. That is, denoting the low-pass filter as LPF, we assume that $f \approx \mathsf{LPF}\{f + w\}$. Such a low-pass filter should have a zero-phase frequency response (otherwise phase distortion precludes an accurate approximation).

Given the signal y in (12), we aim to estimate the noise-free signal, i.e., $x = f + g$. We note that the approach taken here does not ultimately estimate f and g individually. There is an inherent non-uniqueness regarding f and g: they are each defined only up to an additive piecewise polynomial signal (with polynomial segments of degree $K-1$). This is because a piecewise polynomial signals is both low-pass and has a sparse order-K derivative. By estimating the sum, $f + g$, instead of f and g individually, we avoid this ambiguity and improve the conditioning of the estimation problem.

Our approach to estimate x ($x = f + g$) from y is based on the low-pass filter, LPF, which we assume is known. Note that, if an estimate, $\hat{g}$ were available, then we may estimate f by low-pass filtering; i.e., $\hat{f} = \mathsf{LPF}\{y - \hat{g}\}$. Then an estimate, $\hat{x}$, is given by

$$\begin{aligned} \hat{x} &= \hat{f} + \hat{g} && (13)\\ &= \mathsf{LPF}\{y - \hat{g}\} + \hat{g} && (14)\\ &= \mathsf{LPF}\{y\} + \mathsf{HPF}\{\hat{g}\} && (15) \end{aligned}$$

where HPF is the high-pass filter defined by $\mathsf{HPF} = \mathsf{I} - \mathsf{LPF}$. Here, I is the identity operator.

It remains to estimate g. We assumed above that $f \approx \mathsf{LPF}\{f\}$, therefore we have $\mathsf{HPF}\{f\} \approx 0$. Consequently, applying the high-pass filter to (12), we obtain

$$\mathsf{HPF}\{y - \hat{g}\} \approx w. \tag{16}$$

Equation (16) implies that an accurate estimate of g is one that, when subtracted from the data y, yields a signal similar to noise, subsequent to high-pass filtering. Formulating the estimation of g as a linear inverse problem, (16) suggests the data fidelity term should have the form $\|\mathbf{H}(\mathbf{y} - \mathbf{g})\|_2^2$ where $\mathbf{H}$ is the high-pass filter matrix.

3.1 Cost function

Based on the foregoing discussion, we formulate the estimation of g as a penalized least squares problem. The penalty term is chosen to promote the behavior of g that we have assumed; i.e., the order-K derivative of g is sparse. Specifically, we formulate SASS as the problem

$$\mathbf{g}^* = \arg\min_{\mathbf{g}} \frac{1}{2}\|\mathbf{H}(\mathbf{y}-\mathbf{g})\|_2^2 + \lambda \sum_n \phi([\mathbf{D}\mathbf{g}]_n) \tag{17}$$

where $\lambda > 0$, $\mathbf{g} \in \mathbb{R}^N$, and $\mathbf{D}$ is the order-K difference matrix. In (17), the notation $[\mathbf{v}]_n$ denotes the n-th component of vector $\mathbf{v}$.

We take $\mathbf{H}$ in to be a high-pass filter of the form $\mathbf{H} = \mathbf{A}^{-1}\mathbf{B}$ where $\mathbf{A}$ and $\mathbf{B}$ are banded and where, furthermore, $\mathbf{B} = \mathbf{B}_1\mathbf{D}$ where $\mathbf{D}$ is the order-K difference matrix and $\mathbf{B}_1$ is banded (cf. (9) and (11)). Using the high-pass filter in (10), these conditions are satisfied when $K \leq 2d$.

The penalty function $\phi : \mathbb{R} \to \mathbb{R}$ is chosen so as to promote sparsity. Examples of sparsity-promoting functions include

$$\phi(u) = |u|, \quad \text{and} \quad \phi(u) = \frac{1}{a}\log(1+a|u|),\ a > 0. \tag{18}$$

Numerous other penalty functions have also been utilized for sparse optimization [10, 42]. If $\phi(u)$ is convex, then the optimization problem (17) is convex.

The formulation (17) generalizes the LPF/TVD problem in [53] to the higher order case ($K > 1$) and to more general (non-convex) penalty functions.

3.2 Change of variables

We use a change of variables that simplifies problem (17) in two ways. First, note that the value of the cost function in (17) is unaffected if a polynomial of degree $K-1$ is added to g. This is because $\mathbf{D}$ annihilates such polynomials, as does $\mathbf{H}$ (because $\mathbf{D}$ is a right factor of $\mathbf{H}$). Hence, the minimizer of (17) is unique only up to an additive polynomial. Note in addition, that the penalty term in (17) is non-separable; i.e., the elements of g are coupled. This coupling complicates the optimization problem and optimality conditions.

Both issues are eliminated by the change of variables,

$$\mathbf{u} = \mathbf{D}\mathbf{g}, \quad \mathbf{u} \in \mathbb{R}^{N-K},\ \mathbf{g} \in \mathbb{R}^N \tag{19}$$

where $\mathbf{D}$ is the order-K difference matrix of size $(N-K) \times N$. Note that

$$\mathbf{H}\mathbf{g} = \mathbf{A}^{-1}\mathbf{B}\mathbf{g} = \mathbf{A}^{-1}\mathbf{B}_1\mathbf{D}\mathbf{g} = \mathbf{A}^{-1}\mathbf{B}_1\mathbf{u}. \tag{20}$$

Hence, the optimization problem (17) becomes

$$\mathbf{u}^* = \arg\min_{\mathbf{u}} \Big\{ F(\mathbf{u}) = \frac{1}{2}\|\mathbf{H}\mathbf{y} - \mathbf{A}^{-1}\mathbf{B}_1\mathbf{u}\|_2^2 + \lambda \sum_n \phi(u(n)) \Big\}. \tag{21}$$

Note that the cost function, $F : \mathbb{R}^{N-K} \to \mathbb{R}$, is non-differentiable.

The change of variables (19) effectively eliminates $\mathbf{D}$ from both the penalty term and $\mathbf{H}$. As a result, the solution $\mathbf{u}^*$ is unique and the elements of $\mathbf{u}$ in the penalty term of (21) are decoupled.

Note that, given $\mathbf{u}$, the signal $\mathbf{g}$ can not be uniquely determined by (19). Hence, we are unable to accurately determine $\mathbf{g}$. However, as we now show, this poses no issue in the estimation of the sum, $\mathbf{x} = \mathbf{f} + \mathbf{g}$. From (15), we estimate $\mathbf{x}$ as

$$\hat{\mathbf{x}} = \mathbf{L}\mathbf{y} + \mathbf{H}\mathbf{g}. \tag{22}$$

where $\mathbf{L}$ is the low-pass filter matrix, $\mathbf{L} = \tilde{\mathbf{I}} - \mathbf{H}$. Note that, because $\mathbf{H}$ is not quite square, $\tilde{\mathbf{I}}$ can not be exactly the identity matrix. Instead, $\tilde{\mathbf{I}}$ is obtained from the identity matrix by removing the first and last few rows [53]. The matrix $\tilde{\mathbf{I}}$ truncates a signal to be compatible with $\mathbf{H}$. Using (20), we have

$$\hat{\mathbf{x}} = \mathbf{L}\mathbf{y} + \mathbf{A}^{-1}\mathbf{B}_1\mathbf{u} \tag{23}$$

where $\mathbf{u}$ is the sparse signal obtained by minimizing F. Hence, we estimate $\mathbf{x}$ (i.e., $\mathbf{f} + \mathbf{g}$) without estimating $\mathbf{f}$ and $\mathbf{g}$ individually.

The change of variables (19) was also used in [53] for the special case $K = 1$. However, in that work, the component g was explicitly estimated by integration. For $K > 1$, that procedure is very unstable as it leads to K-order integration. The SASS method solves that problem by avoiding the explicit estimation of g; instead, the sum, $\hat{x} = \hat{f} + \hat{g}$, is estimated (using (23)).

3.3 Optimality condition

When ϕ is convex, then $\mathbf{u}^* \in \mathbb{R}^{N-K}$ minimizes F in (21) if and only if

$$\mathbf{0} \in \partial F(\mathbf{u}^*) \tag{24}$$

where ∂F is the subgradient of F [26]. The subgradient of F is given by

$$\partial F(\mathbf{u}) = \mathbf{H_1}^{\mathsf{T}}(\mathbf{A}^{-1}\mathbf{B}_1\mathbf{u} - \mathbf{H}\mathbf{y}) + \lambda\, \partial\phi(\mathbf{u}) \tag{25}$$

where $\mathbf{H}_1 = \mathbf{A}^{-1}\mathbf{B}_1$. So, the optimality condition (24) can be written as

$$\frac{1}{\lambda}\, \mathbf{B}_1^{\mathsf{T}}(\mathbf{A}\mathbf{A}^{\mathsf{T}})^{-1}(\mathbf{B}\mathbf{y} - \mathbf{B}_1\mathbf{u}^*) \in \partial\phi(\mathbf{u}^*). \tag{26}$$

For ℓ_1 norm regularization, i.e., $\phi(u) = |u|$, we have $\partial\phi(u) = \mathrm{sign}(u)$ where sign is the set-valued function,

$$\mathrm{sign}(u) = \begin{cases} \{1\}, & u > 0 \\ [-1, 1], & u = 0 \\ \{-1\}, & u < 0. \end{cases} \tag{27}$$

In this case, the optimality condition (26) can be written as

$$\frac{1}{\lambda}\left[\mathbf{B}_1^{\mathsf{T}}(\mathbf{A}\mathbf{A}^{\mathsf{T}})^{-1}(\mathbf{B}\mathbf{y} - \mathbf{B}_1\mathbf{u}^*)\right]_n \begin{cases} \in [-1,\ 1], & u^*(n) = 0 \\ = 1, & u^*(n) > 0 \\ = -1, & u^*(n) < 0 \end{cases} \tag{28}$$

for all $0 \leqslant n \leqslant N-K-1$. This condition can be used to easily validate the optimality of a solution produced by a numerical algorithm. It can also be used to gauge the convergence of an algorithm minimizing F.

This optimality condition can be illustrated graphically as a scatter plot. The values in (28), when plotted versus $u(n)$, must lie on the graph of the sign function; for example, see Fig. 4(a). We used such a scatter plot also in [53] to verify optimality in the convex case (i.e., ℓ_1 norm). Here, we use a scatter plot of this form also in the non-convex case, to verify that the solution is (locally) optimal (see Fig. 4(b)); and more importantly in the non-convex case, to identify and correct falsely-locked zeros arising in the optimization process (see Fig. 8), as described in Sec. 4.3.

3.4 Setting $\boldsymbol{\lambda}$

The solution $\mathbf{u}^*$ depends significantly on the regularization parameter, λ. Several methods can be used to set λ. Following [27], we describe a simple method for setting λ. The derivation is based on the optimality condition (28). We used this approach in [53] for the convex case; here we use it also for the non-convex case, as described in Sec. 4.2, which is justified if the non-convex regularizer is suitably constrained [52].

Suppose, in some realization of $\mathbf{y}$ in (12), that $\mathbf{g}$ is identically zero. In this case, we hope that $\mathbf{u}^*$ is also identically zero. Equivalently, using (28),

$$\frac{1}{\lambda}\,\left[\mathbf{B}_1^{\mathsf{T}}(\mathbf{A}\mathbf{A}^{\mathsf{T}})^{-1}\mathbf{B}\mathbf{y}\right]_n \in [-1,\ 1], \quad \forall n \tag{29}$$

with $\mathbf{y} = \mathbf{f} + \mathbf{w}$, where $\mathbf{f}$ and $\mathbf{w}$ are the low-pass signal and the additive noise signal, respectively ($\mathbf{g}$ being zero). Note that the matrix in (29) incorporates the high-pass filter $\mathbf{H}$ as a factor; hence $\mathbf{B}_1^{\mathsf{T}}(\mathbf{A}\mathbf{A}^{\mathsf{T}})^{-1}\mathbf{B}\mathbf{f} \approx 0$ because the high-pass filter annihilates the low-pass signal, $\mathbf{f}$. Therefore, replacing $\mathbf{y}$ in (29) by $\mathbf{w}$, (29) still holds approximately. Hence, (29) suggests that λ be set as

$$\lambda \approx \max_n \left|\left[\mathbf{B}_1^{\mathsf{T}}(\mathbf{A}\mathbf{A}^{\mathsf{T}})^{-1}\mathbf{B}\mathbf{w}\right]_n\right|. \tag{30}$$

The equation (30) can interpreted in the sense of the 'three-sigma' rule, i.e., most observations of a random variable fall within three standard deviations of its mean. Accordingly, we approximate (30) as

$$\lambda \approx 3\,\mathrm{std}\{\mathbf{B}_1^{\mathsf{T}}(\mathbf{A}\mathbf{A}^{\mathsf{T}})^{-1}\mathbf{B}\mathbf{w}\}. \tag{31}$$

We define $\mathbf{r}$ as the random signal $\mathbf{r} = \mathbf{B}_1^{\mathsf{T}}(\mathbf{A}\mathbf{A}^{\mathsf{T}})^{-1}\mathbf{B}\mathbf{w}$. If the noise $\mathbf{w}$ is zero-mean white Gaussian with standard deviation σ, then, disregarding start and end transients, the signal $\mathbf{r}$ is stationary and its standard deviation, σ_r, is given by $\sigma_r = \|\mathbf{p}\|_2\,\sigma$ where $\mathbf{p}$ represents the impulse response of the LTI system $\mathbf{P} = \mathbf{B}_1^{\mathsf{T}}(\mathbf{A}\mathbf{A}^{\mathsf{T}})^{-1}\mathbf{B} = \mathbf{H}_1^{\mathsf{T}}\mathbf{H}$. Hence, in the case of additive white Gaussian noise, (31) can be written as

$$\lambda \approx 3\,\|\mathbf{p}\|_2\,\sigma. \tag{32}$$

This approach sets λ proportional to the noise standard deviation, σ.

4 Optimization Algorithm

In this section, we describe an iterative algorithm for sparsity-assisted signal smoothing (SASS). The algorithm minimizes F in (21). Numerous algorithmic frameworks for sparse optimization can be used, e.g., FOCUSS [47], iterative reweighted least squares (IRLS) [19], generalizations of IRLS [48, 59], ISTA [18, 25], and proximal splitting methods [13, 14], among others.

The SASS algorithm we propose takes advantage of the fact that the filter matrices, $\mathbf{A}$ and $\mathbf{B}$, and the order-K difference matrix, $\mathbf{D}$, are all banded. As a result, the algorithm is computationally efficient because it can be implemented using fast solvers for banded systems. The algorithm does not require the user to specify additional parameters.

4.1 Majorization-minimization

We use the majorization-minimization (MM) optimization framework [23] to develop an iterative algorithm to minimize (21). The MM procedure minimizes a function F by defining an iterative algorithm via

$$\mathbf{u}^{(i+1)} = \arg\min_{\mathbf{u}} G(\mathbf{u}, \mathbf{u}^{(i)}) \tag{33}$$

where i is the iteration index and G is some suitably chosen majorizer of F. Specifically, G should satisfy: $G(\mathbf{u}, \mathbf{v}) \geqslant F(\mathbf{u})\, \forall \mathbf{u}$, and $G(\mathbf{v}, \mathbf{v}) = F(\mathbf{v})$. The MM process is most effective when the chosen majorizer, G, is easily minimized. With initialization $\mathbf{u}^{(0)}$, the update (33) produces a sequence $\mathbf{u}^{(i)}$ converging to a minimizer of F under mild assumptions (or a local minimizer, if F is not convex). Here we use the MM procedure for both convex and non-convex cases. In either case, the MM procedure ensures the cost function decreases each iteration. For more details, see [23] and references therein.

To construct an easily minimized majorizer of F in (21), we define a quadratic majorizer of the penalty term, $\lambda \sum_n \phi(u(n))$. Assuming the penalty function, $\phi(u)$, is concave on $\mathbb{R}_+$, symmetric, and differentiable for $u \neq 0$, such as the those in (18), then a quadratic majorizer can be readily found. Specifically, a majorizer of $\phi(u)$, is given by

$$g(u, v) = \frac{\phi'(v)}{2v}\, u^2 + \phi(v) - \frac{v}{2}\phi'(v), \tag{34}$$

as derived in [50] and illustrated in Fig. 1. That is, for $v \neq 0$,

$$g(u, v) \geqslant \phi(u), \quad \text{for all } u \in \mathbb{R} \tag{35}$$

$$g(v, v) = \phi(v). \tag{36}$$

Note that $g(u, v)$ is quadratic in u.

The majorizer g can be used to obtain a majorizer for F in (21). If $\mathbf{u}$ and $\mathbf{v}$ are vectors, then

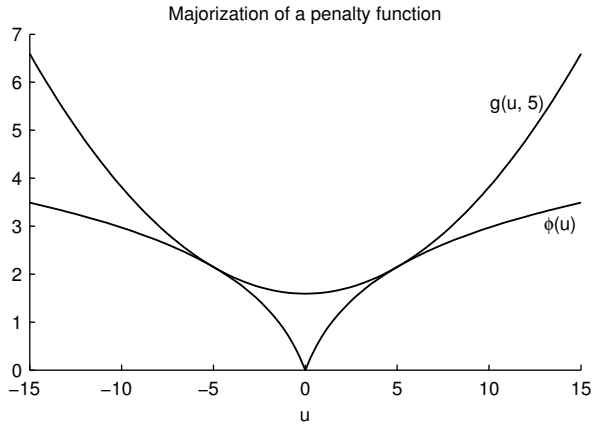

Fig. 1. Majorization of penalty function. The function $g(u, v)$ majorizes the function $\phi(u)$ and is equal to $\phi(u)$ at $u = v$. In the figure, $v = 5$, at which point the two functions are tangent.

$$\lambda \sum_n g(u(n), v(n)) \geqslant \lambda \sum_n \phi(u(n)) \tag{37}$$

with equality if $\mathbf{u} = \mathbf{v}$. That is, the left-hand-side of (37) is a majorizer of $\lambda \sum_n \phi(u(n))$. Moreover, the left-hand-side of (37) can be written compactly as

$$\lambda \sum_n g(u(n), v(n)) = \frac{1}{2} \mathbf{u}^\mathsf{T} [\mathbf{\Lambda}(\mathbf{v})]^{-1} \mathbf{u} + c \tag{38}$$

where $\mathbf{\Lambda}(\mathbf{v})$ is a diagonal matrix defined by

$$[\mathbf{\Lambda}(\mathbf{v})]_{n,n} = \frac{1}{\lambda} \frac{v(n)}{\phi'(v(n))} \tag{39}$$

and c is a constant that does not depend on $\mathbf{u}$. Therefore, using (37), a majorizer of F in (21) is given by

$$G(\mathbf{u}, \mathbf{v}) = \frac{1}{2} \|\mathbf{H}\mathbf{y} - \mathbf{A}^{-1}\mathbf{B}_1\mathbf{u}\|_2^2 + \frac{1}{2} \mathbf{u}^\mathsf{T} [\mathbf{\Lambda}(\mathbf{v})]^{-1} \mathbf{u} + c.$$

$G(\mathbf{u}, \mathbf{v})$ is quadratic in $\mathbf{u}$. Hence, minimizing $G(\mathbf{u}, \mathbf{v})$ with respect to $\mathbf{u}$ can be achieved by setting the gradient to zero, and solving the resulting linear system. Hence, the MM update equation, (33), leads to

$$\mathbf{u}^{(i+1)} = \arg\min_{\mathbf{u}} G(\mathbf{u}, \mathbf{u}^{(i)}) \tag{40}$$

$$= \left(\mathbf{B}_1^\mathsf{T} (\mathbf{A}\mathbf{A}^\mathsf{T})^{-1} \mathbf{B}_1 + [\mathbf{\Lambda}(\mathbf{u}^{(i)})]^{-1} \right)^{-1} \mathbf{B}_1^\mathsf{T} (\mathbf{A}\mathbf{A}^\mathsf{T})^{-1} \mathbf{B}\mathbf{y} \tag{41}$$

where $\mathbf{\Lambda}(\mathbf{u}^{(i)})$ depends on $\mathbf{u}^{(i)}$ per (39).

There are two numerical complications with (41). First, it calls for the solution to a large $(N \times N)$ dense system of equations, which is computationally costly. Secondly, the elements of the diagonal matrix $\mathbf{\Lambda}$ go to zero as $\mathbf{u}^{(i)}$ converges to a sparse vector. Therefore, many 'divide-by-zero' errors arise in practice when implementing (41) directly.

To avoid both problems, the matrix inverse lemma (MIL) can be used as described in [24]. Using the MIL, the inverse of the system matrix can be rewritten as:

$$\begin{aligned} &\left(\mathbf{B}_1^{\mathsf{T}}(\mathbf{A}\mathbf{A}^{\mathsf{T}})^{-1}\,\mathbf{B}_1 + [\mathbf{\Lambda}^{(i)}]^{-1}\right)^{-1} \\ &\quad = \mathbf{\Lambda}^{(i)} - \mathbf{\Lambda}^{(i)}\mathbf{B}_1^{\mathsf{T}}\left(\mathbf{A}\mathbf{A}^{\mathsf{T}} + \mathbf{B}_1\mathbf{\Lambda}^{(i)}\mathbf{B}_1^{\mathsf{T}}\right)^{-1}\mathbf{B}_1\mathbf{\Lambda}^{(i)} \end{aligned} \tag{42}$$

where we use the abbreviation $\mathbf{\Lambda}^{(i)} := [\mathbf{\Lambda}(\mathbf{u}^{(i)})]$, i.e.,

$$[\mathbf{\Lambda}^{(i)}]_{n,n} = \frac{1}{\lambda}\frac{u^{(i)}(n)}{\phi'(u^{(i)}(n))}. \tag{43}$$

Therefore, (41) can be implemented as:

$$\mathbf{b} = \mathbf{B}_1^{\mathsf{T}}(\mathbf{A}\mathbf{A}^{\mathsf{T}})^{-1}\,\mathbf{B}\mathbf{y} \tag{44}$$

$$\mathbf{Q}^{(i)} = \mathbf{A}\mathbf{A}^{\mathsf{T}} + \mathbf{B}_1\mathbf{\Lambda}^{(i)}\mathbf{B}_1^{\mathsf{T}} \tag{45}$$

$$\mathbf{u}^{(i+1)} = \mathbf{\Lambda}^{(i)}[\mathbf{b} - \mathbf{B}_1^{\mathsf{T}}[\mathbf{Q}^{(i)}]^{-1}\mathbf{B}_1\mathbf{\Lambda}^{(i)}\mathbf{b}] \tag{46}$$

Note that the matrices, $\mathbf{A}\mathbf{A}^{\mathsf{T}}$ in (44) and $\mathbf{Q}^{(i)}$ in (45), are banded. Therefore, the inverses in (44) and (46) can be implemented very efficiently using fast solvers for banded systems [46, Sect 2.4]. The complexity of these algorithms are linear in N. In addition, all other matrices appearing in (44)-(46) are diagonal or banded; hence, the matrix multiplications are also efficiently implemented.

The SASS algorithm is summarized in Table 1. Note that the algorithm does not require the user to specify any parameters, such as a step-size parameter. The only parameters are those that define the objective function, (21), i.e., K, λ, ϕ, $\mathbf{A}$, $\mathbf{B}$.

4.2 Non-convex penalty functions

Non-convex penalty functions can promote sparsity more strongly than convex penalty functions; hence, they can yield superior results in some sparse optimization problems. Generally, non-convex penalty functions lead to non-convex optimization problems (see [52] for exceptions). Consequently, algorithms have been developed specifically for the non-convex case, based on iterative reweighted ℓ_1 and/or least squares [7,37,41,55,57,58], splitting [11,29], and other optimization methods [28, 42]. Several methods target ℓ_0 pseudo-norm minimization, for example, by single best replacement (SBR) [54] or

Table 1. Sparsity-assisted signal smoothing (SASS) algorithm

	Input: $\mathbf{y} \in \mathbb{R}^N, K, \lambda, \phi, \mathbf{A}, \mathbf{B}$	
1.	$\mathbf{b} = \mathbf{B}_1^{\mathsf{T}}(\mathbf{A}\mathbf{A}^{\mathsf{T}})^{-1}\mathbf{B}\mathbf{y}$	
2.	$\mathbf{u} = \mathbf{D}\mathbf{y}$	(initialization)
	Repeat	
3.	$\mathbf{\Lambda}_{n,n} = \dfrac{1}{\lambda}\dfrac{u(n)}{\phi'(u(n))}$	($\mathbf{\Lambda}$ is diagonal)
4.	$\mathbf{Q} = \mathbf{A}\mathbf{A}^{\mathsf{T}} + \mathbf{B}_1\mathbf{\Lambda}\mathbf{B}_1^{\mathsf{T}}$	($\mathbf{Q}$ is banded)
5.	$\mathbf{u} = \mathbf{\Lambda}[\mathbf{b} - \mathbf{B}_1^{\mathsf{T}}\mathbf{Q}^{-1}\mathbf{B}_1\mathbf{\Lambda}\mathbf{b}]$	(MM update)
	Until convergence	
6.	$\mathbf{x} = \tilde{\mathbf{I}}\mathbf{y} - \mathbf{A}^{-1}\mathbf{B}\mathbf{y} + \mathbf{A}^{-1}\mathbf{B}_1\mathbf{u}$.	
	Output: $\mathbf{x}$, $\mathbf{u}$	

iterative thresholding [3, 36, 45]. Most of these methods could be applied to the minimization of (21).

We apply the SASS algorithm in both the convex and non-convex cases, the case depending on ϕ. Note that ϕ influences the algorithm only in line 3 of Table 1, i.e., equation (43). So, if we define $\psi : \mathbb{R} \to \mathbb{R}$ as

$$\psi(u) := \frac{u}{\phi'(u)}, \tag{47}$$

then ψ encapsulates the role of the penalty function in the algorithm.

Three penalty functions and the corresponding weight functions are summarized in Table 2. The second and third penalties, involving the logarithmic (log) and arctangent (atan) functions, are both non-convex (strictly concave on $\mathbb{R}_+$) and parameterized by the parameter $a > 0$. Increasing a increases the non-convexity of ϕ. The atan penalty, derived in [52] as a natural extension of the log penalty, promotes sparsity more strongly than the log penalty. Compared to the commonly used ℓ_p pseudo-norm with $p < 1$, the log and atan functions have the advantage that for suitably constrained a, the penalized least squares problem (e.g., (21)) can be convex even when the penalty function is not [52]. For the ℓ_p pseudo-norm, this possibility is precluded due to the fact that the derivative of $|u|^p$ goes to infinity at zero for $p < 1$.

Unfortunately, the use of a general non-convex penalty function complicates the setting of the regularization parameter, λ. For the non-convex case, the simple guideline (32) is not valid in general, because that equation is derived based on ℓ_1 norm regularization, i.e., convex regularization. However, if F is convex (even if ϕ is not convex), then the guideline (32) is still valid [52]. Specifically, if the log or atan penalty is used and if a is suitably constrained, then (32) is still useful for setting λ.

Table 2. Sparse penalties and corresponding weight functions

penalty, $\phi(u)$	weight, $\psi(u)$
$\lvert u\rvert$ (i.e., ℓ_1 norm)	$\lvert u\rvert$
$\frac{1}{a}\log(1+a\lvert u\rvert)$	$\lvert u\rvert(1+a\lvert u\rvert)$
$\frac{2}{a\sqrt{3}}\left(\tan^{-1}\left(\frac{1+2a\lvert u\rvert}{\sqrt{3}}\right)-\frac{\pi}{6}\right)$	$\lvert u\rvert(1+a\lvert u\rvert+a^2\lvert u\rvert^2)$

When using the logarithmic (log) and arctangent (atan) penalty functions, how should the parameter, a, be specified? Note that a controls the extent to which the functions are non-convex. As $a \to 0$, the log and atan penalties approach the ℓ_1 norm. For $a > 0$, the log and atan penalties are strictly concave on $\mathbb{R}_+$. An upper bound on a, that guarantees F is convex, can be obtained by semidefinite programming (SDP) [52]. Here, we describe a heuristic to set a, so as to avoid the computational complexity of SDP.

To derive a heuristic to set the non-convexity parameter, a, in the log and atan functions, we make a simplifying assumption. Namely, we assume the sparse vector, $\mathbf{u}^*$, minimizing F, contains only a single non-zero entry. While not satisfied in practice, with this assumption we obtain a value for a above which F is definitely non-convex. Using corollary 1 of [52], this assumption leads to an upper bound on a of $\|\mathbf{h}_1\|_2^2/\lambda$ where $\mathbf{h}_1$ represents the impulse response of the system $\mathbf{H}_1 := \mathbf{A}^{-1}\mathbf{B}_1$. (In computing the impulse response, start and end transients due to signal boundaries should be omitted.) Because the assumption is idealized, this upper bound is expected to be too high (i.e., not guaranteeing convexity of F), hence a smaller, more conservative value of a is appropriate. In the examples below, we use half this value; i.e., we set

$$a = 0.5\,\|\mathbf{h}_1\|_2^2/\lambda. \tag{48}$$

4.3 Zero-locking

The SASS algorithm in Table 1 is susceptible to 'zero-locking' behavior. That is, if a component $u^{(i)}(m)$ is zero on some iteration i for some m, then it will be zero for all subsequent iterations, i.e., $u^{(k)}(m) = 0$ for all $k > i$. The zero is 'locked in'. This occurs because, if $u^{(i)}(m) = 0$, then $[\mathbf{\Lambda}^{(i)}]_{m,m} = 0$, and consequently $u^{(i+1)}(m) = 0$. For this reason, the algorithm should be initialized with non-zeros.

This zero-locking behavior (or 'singularity issue') is a well known phenomenon in reweighted methods for sparse optimization [23, 27, 43]. But it does not necessarily impede the convergence of algorithms in which it occurs [23, 27, 43]. We have found that in the convex case (i.e., ℓ_1 norm regularization), the algorithm converges reliably to an optimal solution in practice. We validate optimality using (28).

In the non-convex case, convergence to only a local optimal solution can be expected. We have found experimentally, that in the non-convex case, the zero-locking issue sometimes causes the algorithm to converge to a solution that is not locally optimal. This can be recognized using condition (26). Graphically, some points in the scatter plot will lie off the graph of $\phi'(u)$, as illustrated in Example 2 (Fig. 8(a)) below.

In this way, we identify those values, $u(n)$, if any, that are incorrectly locked to zero. Those values can then be perturbed, and the SASS algorithm can be run a second time. In our implementation, we perturb these values using least squares. Specifically, we hold the other components of $\mathbf{u}$ fixed, and solve $\mathbf{Hy} \approx \mathbf{A}^{-1}\mathbf{Bu}$ in the least square sense over the components of $\mathbf{u}$ that are identified as incorrectly locked to zero. (In our experiments, there are few, if any, incorrectly locked zeros; hence, the least square problem is small in size, and computationally negligible.) After running the SASS algorithm a second time, we obtain a new sparse vector, $\mathbf{u}$; then we generate a new scatter plot and checked for incorrectly locked zeros. In our experiments, we have found that, if λ is set according to (32), then only a second or third run of SASS is sufficient to correct all falsely locked zeros, when they occur. In Example 2 below, we illustrate the use of this process to overcome the zero-locking issue.

4.4 Preprocessing to avoid start and end transients

The implementation of a recursive digital filter as $\mathbf{H} = \mathbf{A}^{-1}\mathbf{B}$, where $\mathbf{A}$ and $\mathbf{B}$ are banded matrices, can induce undesirable transients at the start and end of the signal. In the examples, we alleviate this issue using a simple preprocessing step. Namely, we perform low-order least-square polynomial approximation on the first and last segments of the noisy data, before using the SASS algorithm. In particular, we replace the first and last 15 points of the noisy signal by polynomials of degree r. In Examples 1 and 2, we use a polynomials of degree 1 and 2, respectively. In Example 1, we use a second-order high-pass filter, $\mathbf{H}$, (with $d = 1$), which perfectly annihilates the degree 1 polynomial. Likewise, in Example 2, we use a fourth-order high-pass filter ($d = 2$) which annihilates the degree 2 polynomial. Hence, transients due to the signal boundaries are avoided. This is effective, provided the signal has no singularities too close to either of its end points.

5 Example 1

This example illustrates sparsity-assisted signal smoothing (SASS) to estimate the piecewise smooth signal, of the form $f+g$, shown in Fig. 2(a). The signal, f, consists of several low-frequency sinusoids; g is piecewise linear.

The noisy signal, $y = f + g + w$, illustrated in Fig. 2(b), has additive white Gaussian noise (AWG) with standard deviation, $\sigma = 0.5$. We set the low-pass filter, $\mathbf{L} = \mathbf{I} - \mathbf{H} = \mathbf{I} - \mathbf{A}^{-1}\mathbf{B}$, to be a second order zero-phase

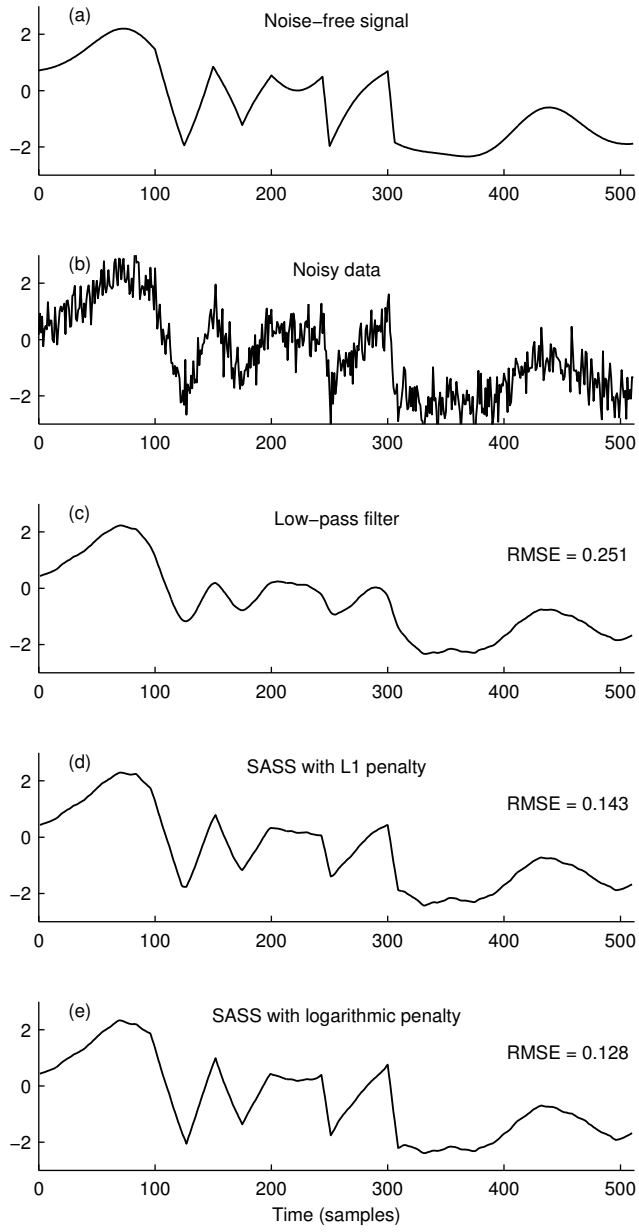

Fig. 2. Example 1. Sparsity-assisted signal smoothing (SASS). (a) Noise-free signal. (b) Noisy data, $\mathbf{y}$. (c) Output of low-pass filter, $\mathbf{Ly}$. (d, e) Output of SASS algorithm with ℓ_1 norm and logarithmic penalties, respectively.

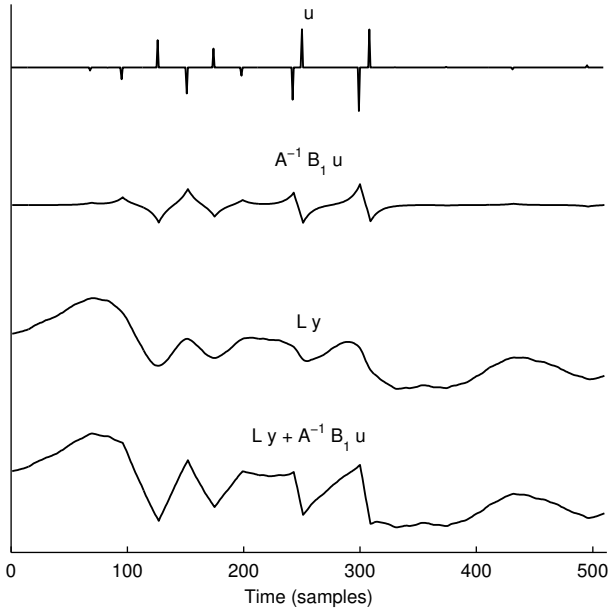

Fig. 3. Example 1. Components of the SASS output shown in Fig. 2(e). The sparse signal, **u**, is obtained by sparse optimization. The SASS output is given by (23).

Butterworth filter ($d = 1$) with a cut-off frequency of $f_c = 0.02$ cycles/sample (10) [53]. The cut-off frequency is set so that the pass-band encompasses the sinusoidal components in f. The output of the low-pass filter, shown in Fig. 2(c), is relatively free of noise. However, parts of the signal are heavily distorted; specifically, those parts where the derivative is discontinuous, due to the piecewise linear component, g.

To use the SASS algorithm, the parameters K and λ must be set. In this example, we set $K = 2$; that is, we model g as having a sparse order-2 derivative. Implicitly, we model g as approximately piecewise linear. To set the regularization parameter, λ, we use (32). For the logarithmic and arctangent penalties, we also need to specify the additional parameter, a, which controls the degree of non-convexity of the penalty function. We set a using (48) as described in Sect. 4.2. In this example, we have run SASS for 100 iterations. The run-time was 36 milliseconds, measured on a 2013 MacBook Pro (2.5 GHz Intel Core i5) running Matlab R2011a.

The output of the SASS algorithm using the ℓ_1 norm penalty is shown in Fig. 2(d). Note that the signal is similar to the low-pass filtered signal, but it exhibits sharp features not possible with low-pass filtering. The RMSE (root-mean-square error) is also substantially better than the RMSE of the low-pass filter. The output of the SASS algorithm using the logarithmic penalty function is shown in Fig. 2(e). The result is similar to that obtained with the

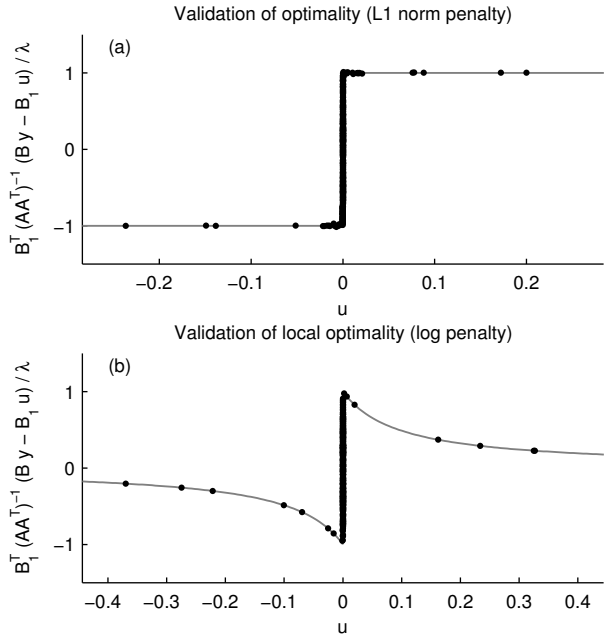

Fig. 4. Example 1: Scatter plots to validate optimality. (a) Optimality of the ℓ_1 norm solution. (b) Local optimality of the logarithmic penalty solution.

ℓ_1 norm, but the sharp features are of somewhat greater amplitude; it also has a lower RMSE. Note that the SASS output signals, in Fig. 2(d, e), are relatively free of Gibbs' phenomenon. There is negligible ringing around the singularities of the signal. This is due, in part, to the SASS approach being based on total variation (TV) denoising, which is free of Gibbs' phenomenon.

We illustrate the SASS solution further in Fig. 3. The sparse vector, $\mathbf{u}$, minimizing F in (21), with the logarithmic penalty, is shown in the figure. Once $\mathbf{u}$ is obtained, we compute $\mathbf{A}^{-1}\mathbf{B}_1\mathbf{u}$ in (23). As shown in the figure, this signal exhibits points where the derivative is discontinuous. The final SASS output is then obtained by adding the low-pass filtered signal, $\mathbf{Ly}$, and $\mathbf{A}^{-1}\mathbf{B}_1\mathbf{u}$, according to (23). Note that the signal, $\mathbf{A}^{-1}\mathbf{B}_1\mathbf{u}$, can be considered an additive correction, or enhancement, obtained via sparse optimization, of the conventional low-pass filter output.

The optimality of the SASS solutions are validated in Fig. 4. Specifically, the points in the scatter plot lie on the graph of $\partial\phi$, which, for the ℓ_1 norm, is the (set-valued) sign function, as illustrated in Fig. 4(a). The preponderance of points on the line, $u = 0$, corresponds to the fact that $\mathbf{u}$ is sparse.

For the logarithmic penalty, which is not convex, the scatter plot can be used to validate locally optimality, only. The points in the scatter plot should lie on the graph of ϕ' for $u \neq 0$, and in the interval $[-1, 1]$ for $u = 0$. For the

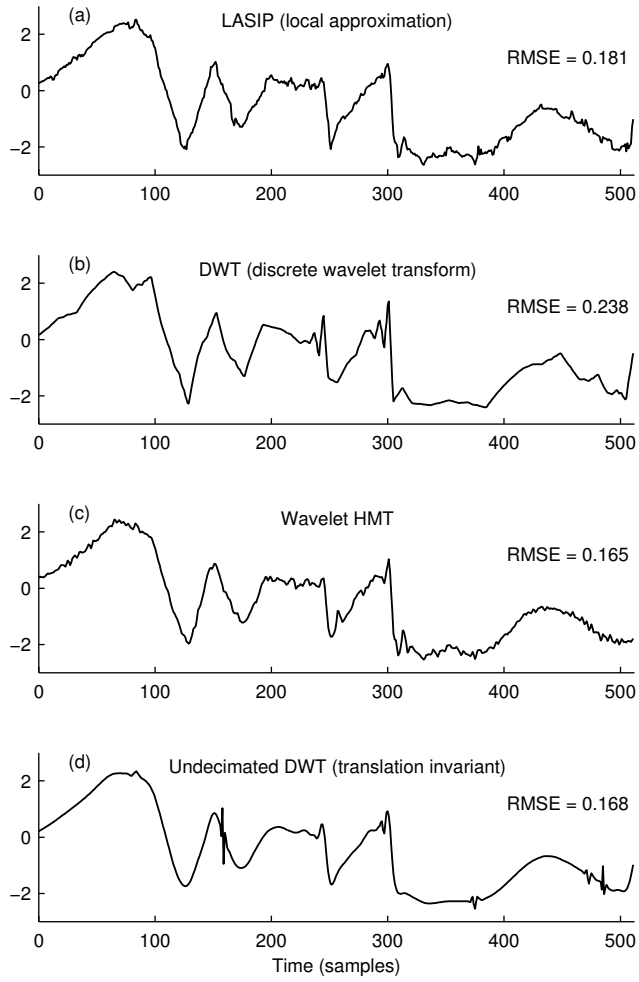

Fig. 5. Example 1. (a) LASIP (local approximation) [35]. (b) DWT (discrete wavelet transform). (c) Wavelet-HMT (hidden Markov tree) [15]. (d) Undecimated wavelet [12]. Compare with Fig. 2.

log penalty, we have

$$\phi'(u) = \frac{1}{a|u|+1}\,\mathrm{sign}(u), \quad u \neq 0. \tag{49}$$

As Fig. 4(b) shows, the scatter plot conforms to this condition.

For the purpose of comparison, Fig. 5 shows the result of several other techniques suitable for piecewise smooth signal denoising. Fig. 5(a) shows the result of LASIP, which is based on local polynomial approximation over adaptively determined windows [35]. Fig. 5(b) shows the result of discrete wavelet

transform hard-thresholding. Fig. 5(c) shows the result of wavelet denoising using a hidden Markov tree (HMT) model [15], which performs substantially better than simple thresholding. Fig. 5(d) shows the result of translation-invariant denoising using an un-decimated wavelet transform (UDWT) [12,38], which also performs well, among wavelet-based methods. For each wavelet method, we used a 5-level transform and the orthonormal Daubechies wavelet with three-vanishing moments [17].

Note that the HMT and UDWT results are qualitatively quite different from each other, although they are based on the same wavelet transform and achieve approximately the same RMSE, which shows the influence of the utilized wavelet-domain model (implicit or explicit). A main issue in obtaining good denoising results using wavelet transforms is the suppression of the pseudo-Gibbs phenomenon which tends to arise in wavelet denoising. This requires using the wavelet transform in conjunction with additional models, penalty terms, or constraints, etc. Since the SASS approach does not utilize wavelet transforms, and is based on a simple, explicit model (c.f. (1)), it is relatively unaffected by the pseudo-Gibbs phenomenon.

6 Example 2 (ECG Denoising)

This example illustrates sparsity-assisted signal smoothing (SASS) on the problem of denoising electrocardiogram (ECG) signals. We use the ECG waveform simulator, ECGSYN [40], to generate synthetic ECG signals, with a sampling rate of 256 samples/second. An example of two seconds of simulated ECG, is shown in Fig. 6(a).

The noisy signal, shown in Fig. 6(b), has AWGN with $\sigma = 0.1$. In this example, we set the filter to be a fourth order zero-phase Butterworth filter ($d = 2$) with a cut-off frequency of $f_c = 0.03$ cycles/sample (10). The output of the low-pass filter, shown in Fig. 6(c), is relatively free of noise; however, the peaks of the two QRS complexes are substantially attenuated. The peak-to-peak (P-P) amplitude of the first QRS complex is indicated in the figure. The peaks can be better preserved by using a low-pass filter with a higher cut-off frequency; however, the filter will then let more noise through.

For the SASS algorithm, we use $K = 3$ in this example, which corresponds to modeling the component g as having a sparse order-3 derivative (i.e., approximately piecewise polynomial with polynomial segments of order 2). We set λ using (32) and a using (48). To obtain the atan solution, we initialize the SASS algorithm with the ℓ_1 norm solution.

Figure 6(d, e) shows the output of the SASS algorithm using both the ℓ_1 norm and the arctangent penalty functions. As can be seen, SASS preserves the QRS waveform much more accurately than the low-pass filter. The ℓ_1 solution has a P-P amplitude of 1.30, almost twice that of the LPF. The atan solution has an even higher P-P amplitude of 1.45. The atan penalty function induces less bias, and promotes sparsity more strongly, than the ℓ_1

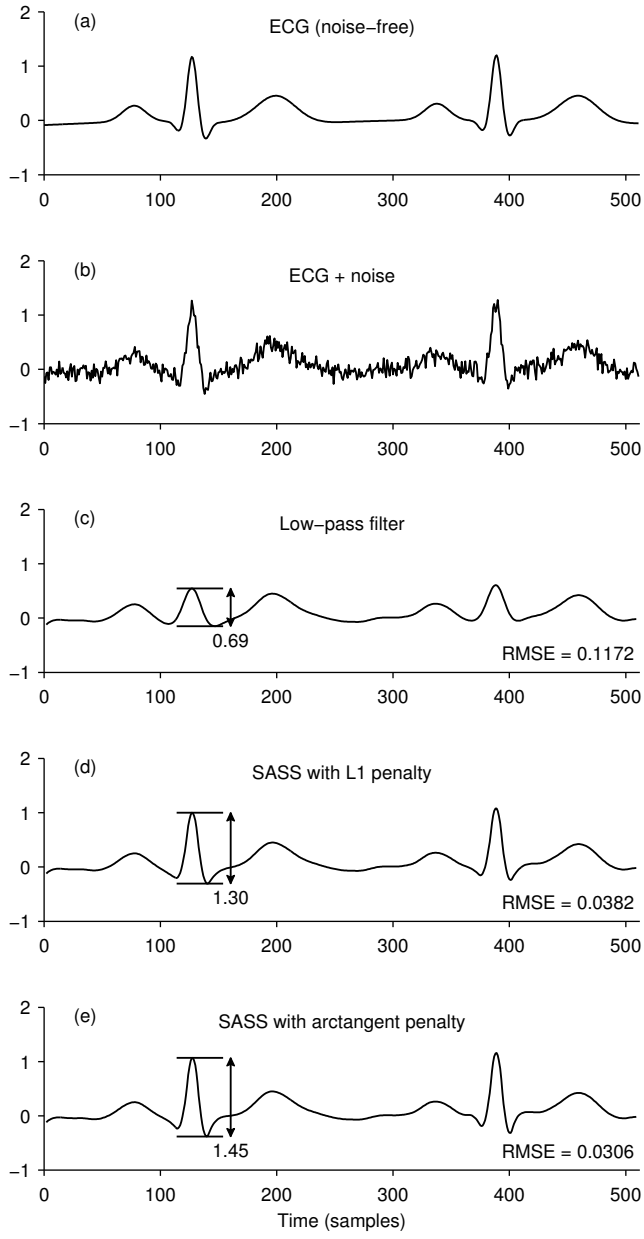

Fig. 6. Example 2. Sparsity-assisted signal smoothing (SASS). (a) Noise-free signal. (b) Noisy data, $\mathbf{y}$. (c) Output of low-pass filter, $\mathbf{Ly}$. (d, e) Output of SASS algorithm with ℓ_1 norm and arctangent penalties, respectively.

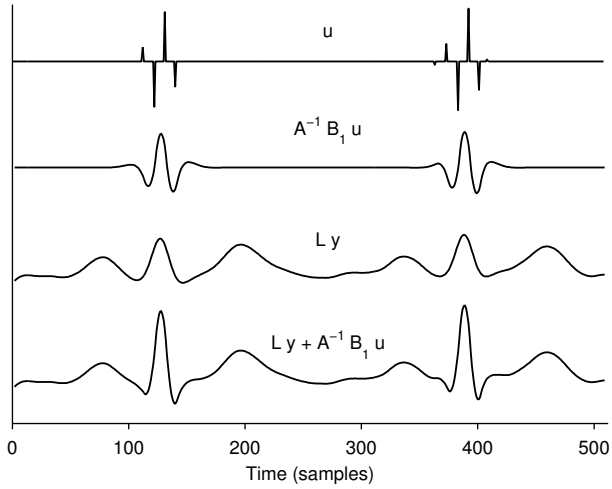

Fig. 7. Example 2. Components of the arctangent solution shown in Fig. 6(e).

norm penalty. The solution obtained with the logarithmic penalty (not shown) is similar to the atan solution (the P-P amplitude of the log solution is 1.43).

We illustrate, in Fig. 7, the components of the SASS solution obtained using the arctangent penalty. As shown, $\mathbf{u}$ is sparse, and the signal, $\mathbf{A}^{-1}\mathbf{B}_1\mathbf{u}$, is composed of a few zero-mean oscillatory waveforms. The final SASS output is obtained by adding the low-pass filtered signal, $\mathbf{L}\mathbf{y}$, and $\mathbf{A}^{-1}\mathbf{B}_1\mathbf{u}$, according to (23).

The optimality scatter plot and zero-correction procedure are illustrated in Fig. 8 for the SASS solution obtained using the arctangent penalty. For the arctangent penalty, which is not convex, the scatter plot can be used to validate locally optimality, only. The points in the scatter plot should lie on the graph of ϕ' for $u \neq 0$, and in $[-1, 1]$ for $u = 0$. For the atan penalty, we have

$$\phi'(u) = \frac{1}{a^2u^2 + a|u| + 1}\,\mathrm{sign}(u), \quad u \neq 0. \tag{50}$$

The output of the SASS algorithm with the arctangent penalty yields the scatter plot shown in Fig. 8(a). Note, Fig. 8(a) shows that four components of $\mathbf{u}$ are positive (dots on the graph of $\phi'(u)$, with $u > 0$). Upon close inspection, as shown in the area of detail, it can be seen that three points lie off the graph, on the line, $u = 0$. These values are incorrectly locked to zero, due to the zero-locking phenomena discussed above. Hence, the SASS algorithm has converged to a solution that is not a local optimum. Having identified, in this way, a set of points falsely locked to zero, we perturb these points away from zero and run the SASS algorithm a second time. The perturbation is performed using least squares. The result of the second run is shown in Fig. 8(b). As can be seen, the points in the scatter plot are entirely on the graph of ϕ'; i.e., no

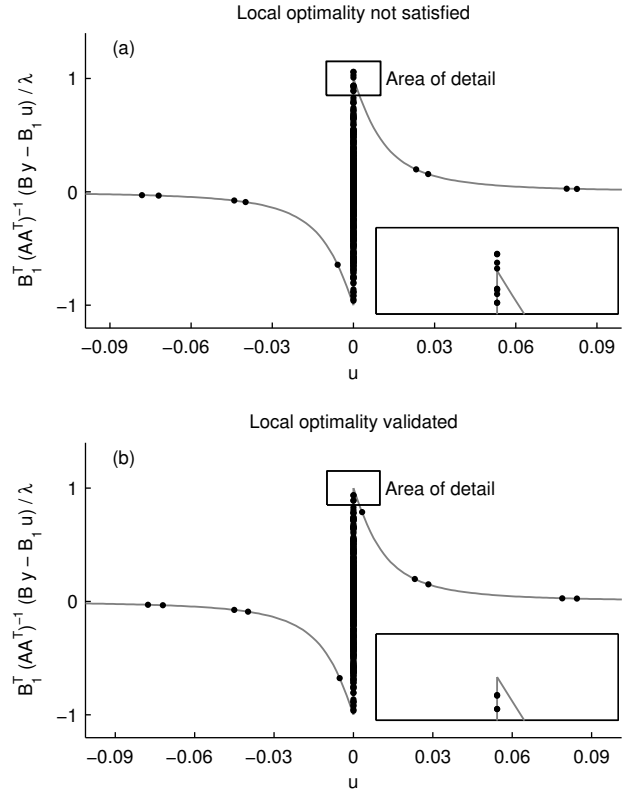

Fig. 8. Example 2: Correction of zero-locking phenomenon. (a) Some points in the scatter plot do not lie on the graph of $\phi'(u)$. The solution is not optimal. (b) After the zero-locking correction procedure, all points in the scatter plot lie on the graph of $\phi'(u)$. The solution is locally optimal.

values are found to be incorrectly locked to zero. Note, Fig. 8(b) shows that five components of $\mathbf{u}$ are positive. That is, one of the components of $\mathbf{u}$ which was incorrectly locked to zero in the first solution, is positive in the second solution. The atan solution shown in Fig. 6(e) is the optimal solution obtained as the result of the second run. We comment that the two atan solutions (local optimum and not) are visually quite similar.

7 Wavelet Functions

The SASS denoising algorithm can be viewed in terms of wavelet denoising. Both methods are based on the same basic signal model; namely, the representation of a signal as the sum of a low-pass (coarse) component and a piecewise smooth (detail) component [6, 17]; i.e., f and g, respectively, in (1).

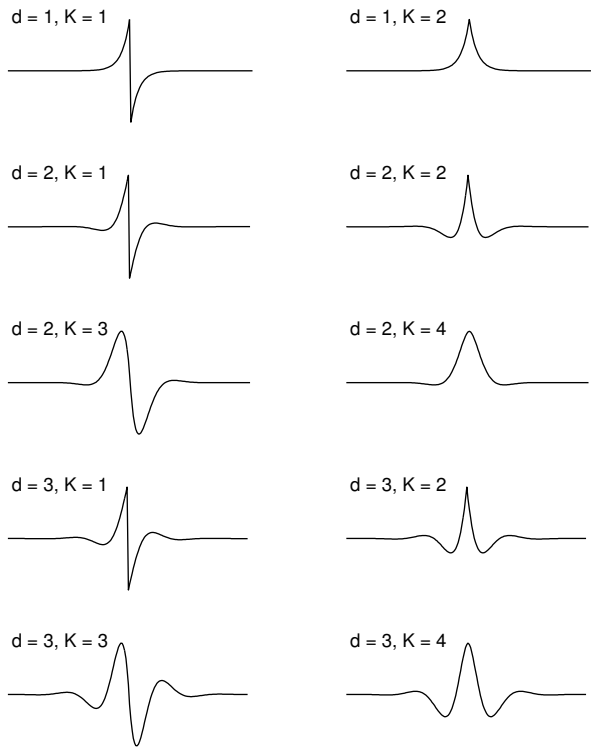

Fig. 9. Impulse response of $\mathbf{H}_1 = \mathbf{A}^{-1}\mathbf{B}_1$ for several values of (d, K), i.e., 'wavelets'.

The similarity can be further illustrated by exhibiting the wavelet-like functions of the SASS algorithm. Note from (23) that the low-pass (coarse) component is given by $\mathbf{Ly}$, i.e., low-pass filtering of the data, $\mathbf{y}$. The piecewise smooth (detail) component is given by $\mathbf{A}^{-1}\mathbf{B}_1\mathbf{u}$, where $\mathbf{u}$ is obtained by sparse optimization. That is, the detail component is a weighted superposition of translations of the impulse response of the filter $\mathbf{H}_1 = \mathbf{A}^{-1}\mathbf{B}_1$. Hence, the impulse response of $\mathbf{H}_1$, denoted $\mathbf{h}_1$, can be considered a 'wavelet'. The detail component, i.e., $\mathbf{H}_1\mathbf{u}$, is a linear combination of translations of the wavelet. In a wavelet transform, the detail component is a linear combination of both translations and *dilations* of the wavelet. Hence, the SASS approach lacks the multiscale properties of wavelet denoising.

It is informative to examine the wavelet, as it determines the characteristics of the detail component. Figure 9 shows the wavelet for several values of the parameters, d and K. In the SASS approach, the parameter, K, is the order of the difference matrix, $\mathbf{D}$. The parameter, d, determines the order of the high-pass filter $\mathbf{H}$; i.e., $\mathbf{H}$ is of order $2d$, see (10). Both d and K should be a small positive integers, with $1 \leqslant K \leqslant 2d$.

As Fig. 9 illustrates, K determines the regularity of the wavelet. For $K = 1$, the wavelet is discontinuous. For $K = 2$, the wavelet is continuous but its derivative is not. For $K = 3$, both the wavelet and its derivative are continuous. The number of vanishing wavelet moments can also be expressed in terms of K and d. Note that the transfer function, $H_1(z)$, has a zero of multiplicity $2d - K$ zeros at $z = 1$; therefore, convolution with the impulse response, $\mathbf{h_1}$, annihilates polynomials of degree $2d-K-1$. Hence, the wavelet can be understood to have $2d-K$ vanishing moments. Note that when $K = 2d$, the wavelet has no vanishing moments; but, as illustrated in Example 1, this does not preclude its effectiveness as it would for a wavelet transform, due to the role of sparse optimization in SASS. The cut-off frequency, f_c, of the filter, $\mathbf{H}$, influences the scale (width) of the wavelet. Varying f_c has the effect of dilating/contracting the wavelet.

Suitable values for d and K, for a particular class of signals, can be chosen based on the characteristics of the wavelet. For example, if it is known that the signals of interest contain additive step discontinuities, then it is reasonable to set $K = 1$, as in [53]. For many signals (e.g., ECG signals), better denoising results are obtained with $K > 1$. (Similarly, wavelets with more than one vanishing moment often produce better results than the Haar wavelet.)

8 Conclusion

We have described a method for signal denoising that utilizes both conventional filtering principles and sparse optimization principles. The method, 'sparsity-assisted signal smoothing' (SASS), is applicable for denoising signals that are piecewise smooth in a general sense, i.e., for which the order-K derivative can be modeled as (approximately) sparse. The SASS approach is based on the formulation of a sparse-regularized linear inverse problem. We provide simple direct approaches to specify the regularization parameter, λ, and the non-convexity parameter, a, the later if a non-convex penalty is utilized. The reweighted least squares algorithm we present, derived using the majorization-minimization principle, is devised so as to maintain the banded property of the involved matrices. Hence, the algorithm is computationally efficient due to the use of fast solvers for banded systems. The optimization algorithm calls for no additional parameters (step sizes, etc.).

The underlying signal model, i.e., a low-pass (coarse) component plus a piecewise smooth (detail) component, also underlies wavelet signal representations. The effectiveness of wavelet-based signal processing is largely due to the sparsity of the wavelet representation of piecewise smooth signals. The proposed SASS approach exploits sparse representations directly via optimization, rather than indirectly through a wavelet transform. Although SASS is not multi-scale, it is relatively free of pseudo-Gibbs phenomena (oscillations around singularities) that often arises in wavelet processing.

Note that the SASS approach will likely be suboptimal for signals having singularities of two (or more) distinct orders (e.g., signals with both additive step discontinuities and 'ramp' discontinuities). The denoising of signals, having singularities of multiple orders, calls for a generalization of SASS.

References

1. R. Al abdi, H. L. Graber, Y. Xu, and R. L. Barbour. Optomechanical imaging system for breast cancer detection. *J. Opt. Soc. Am. A*, 28(12):2473–2493, December 2011.
2. K. Bredies, K. Kunisch, and T. Pock. Total generalized variation. *SIAM J. Imag. Sci.*, 3(3):492–526, 2010.
3. K. Bredies and D. A. Lorenz. Regularization with non-convex separable constraints. *Inverse Problems*, 25(8):085011, 2009.
4. V. Bruni, B. Piccoli, and D. Vitulano. A fast computation method for time scale signal denoising. *Signal Image and Video Processing*, 3(1):63–83, 2008.
5. V. Bruni and D. Vitulano. Wavelet-based signal de-noising via simple singularities approximation. *Signal Processing*, 86(4):859–876, April 2006.
6. C. S. Burrus, R. A. Gopinath, and H. Guo. *Introduction to Wavelets and Wavelet Transforms*. Prentice Hall, 1997.
7. E. J. Candès, M. B. Wakin, and S. Boyd. Enhancing sparsity by reweighted l1 minimization. *J. Fourier Anal. Appl.*, 14(5):877–905, December 2008.
8. A. Chambolle and P.-L. Lions. Image recovery via total variation minimization and related problems. *Numerische Mathematik*, 76:167–188, 1997.
9. T. F. Chan, S. Osher, and J. Shen. The digital TV filter and nonlinear denoising. *IEEE Trans. Image Process.*, 10(2):231–241, February 2001.
10. P. Charbonnier, L. Blanc-Feraud, G. Aubert, and M. Barlaud. Deterministic edge-preserving regularization in computed imaging. *IEEE Trans. Image Process.*, 6(2):298–311, February 1997.
11. R. Chartrand. Fast algorithms for nonconvex compressive sensing: MRI reconstruction from very few data. In *IEEE Int. Symp. Biomed. Imag. (ISBI)*, pages 262–265, July 2009.
12. R. R. Coifman and D. L. Donoho. Translation-invariant de-noising. In A. Antoniadis and G. Oppenheim, editors, *Wavelet and statistics*, pages 125–150. Springer-Verlag, 1995.
13. P. L. Combettes and J.-C. Pesquet. Proximal thresholding algorithm for minimization over orthonormal bases. *SIAM J. on Optimization*, 18(4):1351–1376, 2008.
14. P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In H. H. Bauschke et al., editors, *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*. Springer-Verlag, 2011.
15. M. S. Crouse, R. D. Nowak, and R. G. Baraniuk. Wavelet-based signal processing using hidden Markov models. *IEEE Trans. Signal Process.*, 46(4):886–902, April 1998.
16. V. R. Dantham, S. Holler, V. Kolchenko, Z. Wan, and S. Arnold. Taking whispering gallery-mode single virus detection and sizing to the limit. *Applied Physics Letters*, 101(4):043704, 2012.

17. I. Daubechies. *Ten Lectures On Wavelets*. SIAM, 1992.
18. 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.
19. I. Daubechies, R. DeVore, M. Fornasier, and C. Gunturk. Iteratively reweighted least squares minimization for sparse recovery. *Comm. Pure App. Math.*, 63(1):1–38, January 2010.
20. P. L. Dragotti and M. Vetterli. Wavelet footprints: theory, algorithms, and applications. *IEEE Trans. Signal Process.*, 51(5):1306–1323, May 2003.
21. S. Durand and J. Froment. Reconstruction of wavelet coefficients using total variation minimization. *SIAM J. Sci. Comput.*, 24(5):1754–1767, 2003.
22. S. Durand and M. Nikolova. Denoising of frame coefficients using ℓ^1 data-fidelity term and edge-preserving regularization. *Multiscale Modeling & Simulation*, 6(2):547–576, 2007.
23. 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.
24. M. Figueiredo, J. Bioucas-Dias, J. P. Oliveira, and R. D. Nowak. On total-variation denoising: A new majorization-minimization algorithm and an experimental comparison with wavelet denoising. In *Proc. IEEE Int. Conf. Image Processing*, 2006.
25. M. Figueiredo and R. Nowak. An EM algorithm for wavelet-based image restoration. *IEEE Trans. Image Process.*, 12(8):906–916, August 2003.
26. J.-J. Fuchs. On sparse representations in arbitrary redundant bases. *IEEE Trans. Inform. Theory*, 50(6):1341–1344, 2004.
27. J.-J. Fuchs. Convergence of a sparse representations algorithm applicable to real or complex data. *IEEE. J. Sel. Top. Signal Processing*, 1(4):598–605, December 2007.
28. G. Gasso, A. Rakotomamonjy, and S. Canu. Recovering sparse signals with a certain family of nonconvex penalties and DC programming. *IEEE Trans. Signal Process.*, 57(12):4686–4698, December 2009.
29. A. Gholami and S. M. Hosseini. A general framework for sparsity-based denoising and inversion. *IEEE Trans. Signal Process.*, 59(11):5202–5211, November 2011.
30. A. Gholami and S. M. Hosseini. A balanced combination of Tikhonov and total variation regularizations for reconstruction of piecewise-smooth signals. *Signal Processing*, 93(7):1945–1960, 2013.
31. T.-C. Hsung, D. P. Lun, and W.-C. Siu. Denoising by singularity detection. *IEEE Trans. Signal Process.*, 47(11):3139–3144, 1999.
32. Y. Hu and M. Jacob. Higher degree total variation (HDTV) regularization for image recovery. *IEEE Trans. Image Process.*, 21(5):2559–2571, May 2012.
33. B. Jalil, O. Beya, E. Fauvet, and O. Laligant. Subsignal-based denoising from piecewise linear or constant signal. *Optical Engineering*, 50(11):117004(1–14), November 2011.
34. F. I. Karahanoglu, I. Bayram, and D. Van De Ville. A signal processing approach to generalized 1-d total variation. *IEEE Trans. Signal Process.*, 59(11):5265–5274, November 2011.
35. V. Katkovnik, K. Egiazarian, and J. Astola. *Local Approximation Techniques in Signal and Image Processing*. SPIE Press, 2006.

36. N. Kingsbury and T. Reeves. Redundant representation with complex wavelets: how to achieve sparsity. In *Proc. IEEE Int. Conf. Image Processing*, 2003.
37. I. Kozlov and A. Petukhov. Sparse solutions of underdetermined linear systems. In W. Freeden et al., editor, *Handbook of Geomathematics*. Springer, 2010.
38. M. Lang, H. Guo, J. E. Odegard, C. S. Burrus, and R. O. Wells, Jr. Noise reduction using an undecimated discrete wavelet transform. *IEEE Signal Processing Letters*, 3(1):10–12, January 1996.
39. S.-H. Lee and M. G. Kang. Total variation-based image noise reduction with generalized fidelity function. *IEEE Signal Processing Letters*, 14(11):832–835, November 2007.
40. P. E. McSharry, G. D. Clifford, L. Tarassenko, and L. A. Smith. A dynamical model for generating synthetic electrocardiogram signals. *Trans. on Biomed. Eng.*, 50(3):289–294, March 2003.
41. H. Mohimani, M. Babaie-Zadeh, and C. Jutten. A fast approach for overcomplete sparse decomposition based on smoothed l0 norm. *IEEE Trans. Signal Process.*, 57(1):289–301, January 2009.
42. M. Nikolova, M. K. Ng, and C.-P. Tam. Fast nonconvex nonsmooth minimization methods for image restoration and reconstruction. *IEEE Trans. Image Process.*, 19(12):3073–3088, December 2010.
43. J. Oliveira, J. Bioucas-Dias, and M. A. T. Figueiredo. Adaptive total variation image deblurring: A majorization-minimization approach. *Signal Processing*, 89(9):1683–1693, September 2009.
44. T. W. Parks and C. S. Burrus. *Digital Filter Design*. John Wiley and Sons, 1987.
45. J. Portilla and L. Mancera. L0-based sparse approximation: two alternative methods and some applications. In *Proceedings of SPIE*, volume 6701 (Wavelets XII), 2007.
46. W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical recipes in C: the art of scientific computing (2nd ed.)*. Cambridge University Press, 1992.
47. B. D. Rao, K. Engan, S. F. Cotter, J. Palmer, and K. Kreutz-Delgado. Subset selection in noise based on diversity measure minimization. *IEEE Trans. Signal Process.*, 51(3):760–770, March 2003.
48. P. Rodriguez and B. Wohlberg. Efficient minimization method for a generalized total variation functional. *IEEE Trans. Image Process.*, 18(2):322–332, February 2009.
49. L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.
50. I. Selesnick. Penalty and shrinkage functions for sparse signal processing. *Connexions*, 2012. http://cnx.org/content/m45134/.
51. I. W. Selesnick, S. Arnold, and V. R. Dantham. Polynomial smoothing of time series with additive step discontinuities. *IEEE Trans. Signal Process.*, 60(12):6305–6318, December 2012.
52. I. W. Selesnick and I. Bayram. Sparse signal estimation by maximally sparse convex optimization. *IEEE Trans. Signal Process.*, 62(5):1078–1092, March 2014.
53. I. W. Selesnick, H. L. Graber, D. S. Pfeil, and R. L. Barbour. Simultaneous low-pass filtering and total variation denoising. *IEEE Trans. Signal Process.*, 62(5):1109–1124, March 2014.

54. C. Soussen, J. Idier, D. Brie, and J. Duan. From Bernoulli-Gaussian deconvolution to sparse signal restoration. *IEEE Trans. Signal Process.*, 59(10):4572–4584, October 2011.
55. X. Tan, W. Roberts, J. Li, and P. Stoica. Sparse learning via iterative minimization with application to MIMO radar imaging. *IEEE Trans. Signal Process.*, 59(3):1088–1101, March 2011.
56. D. Van De Ville, B. Forster-Heinlein, M. Unser, and T. Blu. Analytical footprints: Compact representation of elementary singularities in wavelet bases. *IEEE Trans. Signal Process.*, 58(12):6105–6118, 2010.
57. Y. Wang and W. Yin. Sparse signal reconstruction via iterative support detection. *SIAM J. Imag. Sci.*, 3(3):462–491, 2010.
58. D. Wipf and S. Nagarajan. Iterative reweighted ℓ_1 and ℓ_2 methods for finding sparse solutions. *IEEE. J. Sel. Top. Signal Processing*, 4(2):317–329, April 2010.
59. B. Wohlberg and P. Rodriguez. An iteratively reweighted norm algorithm for minimization of total variation functionals. *IEEE Signal Processing Letters*, 14(12):948–951, 2007.