SPARSITY-ASSISTED SIGNAL SMOOTHING (REVISITED)

Ivan Selesnick

Electrical and Computer Engineering
Tandon School of Engineering, New York University
Brooklyn, New York, USA

ABSTRACT

This paper proposes an improved formulation of sparsity-assisted signal smoothing (SASS). The purpose of SASS is to filter/denoise a signal that has jump discontinuities in its derivative (of some designated order) but is otherwise smooth. SASS unifies conventional low-pass filtering and total variation denoising. The SASS algorithm depends on the formulation, in terms of banded Toeplitz matrices, of a zero-phase recursive discrete-time filter as applied to finite-length data. The improved formulation presented in this paper avoids the unwanted end-point transient artifacts which sometimes occur in the original version. For illustration, SASS is applied to ECG signal denoising.

***Index Terms*—** low-pass filter, total variation, sparse signal, denoising, electrocardiogram.

1. INTRODUCTION

Numerous signals can be modeled as the sum of (1) a low-frequency signal and (2) a signal with a sparse K-order derivative. For example, an electrocardiogram (ECG) time-series can be modeled this way. A signal of this kind has jump discontinuities in its derivative (of order K–1) but is otherwise smooth. For the purpose of suppressing additive white Gaussian noise, conventional linear time-invariant (LTI) filters are not suitable for such a signal; LTI filtering tends to over smooth discontinuities (e.g., 'corners' of a signal). Similarly, (generalized) total variation (TV) denoising [29], which is intended for the denoising of piecewise constant (polynomial) signals, is also not suitable for such a signal; TV denoising tends to introduce staircase artifacts.

Sparsity-assisted signal smoothing (SASS) [31] was developed for the purpose of filtering a signal which has discontinuities in its derivative (of some designated order) but is otherwise smooth. SASS combines and unifies conventional LTI low-pass filtering and (generalized) TV denoising. Hence, SASS is useful for a wider class of signals than either LTI low-pass filtering or TV denoising is alone. The SASS algorithm formulates the denoising problem as a sparse deconvolution problem, and in turn, as an optimization problem comprising a data fidelity term and a sparse regularization term. The SASS problem formulation is expressed in terms of banded Toeplitz matrices. Furthermore, the computationally efficient implementation of SASS relies on fast solvers for banded systems of linear equations.

In this paper, we introduce an improved version of SASS.[1] Specifically, we improve the formulation, in terms of banded Toeplitz matrices, of a class of recursive discrete-time filters as applied to finite-length data [31, 34]. This formulation is central to SASS because it allows linear low-pass filtering and nonlinear sparsity-based TV denoising to be combined in a single cost function to be minimized. However, the original formulation [31] gives rise to unwanted transient artifacts at the start and end of finite-length data. Avoiding those artifacts requires ad-hoc preprocessing of the input data which limits the applicability of SASS, especially for the filtering of short input data. The new matrix formulation presented in Sec. 2 below does not give rise to transient artifacts, hence no ad-hoc preprocessing is needed and SASS can be effectively applied to both short and long finite-length data.

1.1. Relation to Prior Work

Several prior works have studied the signal model considered here, i.e., a signal comprising the sum of low-frequency signal and a sparse-derivative signal [11, 16, 26, 32, 33, 34]. The most closely related relevant work is by Gholami and Hosseini [16] who combine Tikhonov (quadratic) regularization and TV denoising. In contrast to [16], SASS is formulated explicitly in terms of an LTI low-pass filter to which it reduces as a special case, and hence can be understood in terms of its frequency response; whereas, the method of [16] is formulated in terms of Tikhonov regularization. In addition, SASS is formulated to allow a higher-order sparse derivative and exploits a factorization (see (30) below) without which the estimated sparse-derivative signal component tends to be unbounded in the higher-order case, which hinders the usability of the result and impedes numerical stability of optimization algorithms. We note that Ref. [16] considers also the problems of deconvolution and compressed sensing, which are not considered here.

As SASS can be considered an extension of TV denoising, we note that several extensions of TV denoising have been proposed [3, 21, 23, 24]. In contrast to these methods, SASS can also be considered an extension of LTI filtering. SASS unifies TV denoising and LTI filtering, and hence conforms to and builds upon elementary signal processing.

Wavelet-based denoising is suitable for the type of piecewise-smooth signal considered here. But simple wavelet-domain denoising leads to artifacts; hence, wavelet-based signal models have been developed to explicitly account for singularities (i.e., discontinuities in the derivative of a signal), such as: hidden Markov tree [9], singularity detection [20, 22], wavelet footprints [12, 35], TV-wavelet [13, 14], and singularity approximation [4, 5]. Although SASS is less general than wavelet-based methods (it does not have a multiscale property), it is much simpler than wavelet-based methods because it does not involve wavelet-domain singularity modeling. SASS preserves sparse singularities in an otherwise smooth signal without inducing wavelet-like artifacts.

This material is based upon work supported by the National Science Foundation under Grant No. 1525398.

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

2. FILTERS AS MATRICES

Given finite-length sequences p_n and q_n, we define banded Toeplitz matrices $\mathbf{P}$ and $\mathbf{Q}$ to have the form

$$\mathbf{P} = \begin{bmatrix} p_2 & p_1 & p_0 & & & \\ & p_2 & p_1 & p_0 & & \\ & & \ddots & & \ddots & \\ & & & p_2 & p_1 & p_0 \end{bmatrix} \tag{1}$$

$$\mathbf{Q} = \begin{bmatrix} q_2 & q_1 & q_0 & & & \\ & q_2 & q_1 & q_0 & & \\ & & \ddots & & \ddots & \\ & & & q_2 & q_1 & q_0 \end{bmatrix}. \tag{2}$$

By the relation between Toeplitz matrices and convolution, we have

$$[\mathbf{Px}]_n = (p * x)(n) \tag{3}$$

where $\mathbf{x}$ is vector containing values $x(n)$ and $\mathbf{P}$ is appropriately indexed. Hence, the matrix $\mathbf{P}$ corresponds to an LTI system with transfer function

$$P(z) = \sum_n p_n z^{-n} \tag{4}$$

and frequency response $P(\mathrm{e}^{\mathrm{j}\omega})$. The matrix $\mathbf{Q}$ corresponds likewise to an LTI system.

Consider the cost function

$$J(\mathbf{x}) = \|\mathbf{Q}(\mathbf{y}-\mathbf{x})\|_2^2 + \alpha\|\mathbf{Px}\|_2^2 \tag{5}$$

with $\alpha > 0$. The function J is minimized by

$$\mathbf{x} = (\mathbf{Q}^{\mathsf{T}}\mathbf{Q} + \alpha\mathbf{P}^{\mathsf{T}}\mathbf{P})^{-1}\mathbf{Q}^{\mathsf{T}}\mathbf{Q}\,\mathbf{y} \tag{6}$$

which constitutes a linear filter with matrix

$$\mathbf{H} = (\mathbf{Q}^{\mathsf{T}}\mathbf{Q} + \alpha\mathbf{P}^{\mathsf{T}}\mathbf{P})^{-1}\mathbf{Q}^{\mathsf{T}}\mathbf{Q}. \tag{7}$$

We define

$$\mathbf{A} := \mathbf{Q}^{\mathsf{T}}\mathbf{Q} + \alpha\mathbf{P}^{\mathsf{T}}\mathbf{P} \tag{8}$$

where we note that $\mathbf{A}$ is *banded*. We thus write $\mathbf{H}$ as

$$\mathbf{H} = \mathbf{A}^{-1}\mathbf{Q}^{\mathsf{T}}\mathbf{Q}. \tag{9}$$

Note that $\mathbf{H}$ is not banded, even though $\mathbf{A}$ and $\mathbf{Q}$ are. Fortunately, the efficient implementation of (6) requires only $\mathbf{A}$ and $\mathbf{Q}$ be banded.

The matrix $\mathbf{H}$ is approximately Toeplitz; thus, it represents (approximately) an LTI system, the frequency response of which is

$$H(\mathrm{e}^{\mathrm{j}\omega}) = \frac{|Q(\mathrm{e}^{\mathrm{j}\omega})|^2}{|Q(\mathrm{e}^{\mathrm{j}\omega})|^2 + \alpha|P(\mathrm{e}^{\mathrm{j}\omega})|^2}. \tag{10}$$

Note that $H(\mathrm{e}^{\mathrm{j}\omega})$ is zero-phase (i.e., real-valued). The transfer function is given by

$$H(z) = \frac{Q(z)Q(1/z)}{Q(z)Q(1/z) + \alpha P(z)P(1/z)}. \tag{11}$$

Expression (6) implements a zero-phase recursive discrete-time filter for finite-length data. It can be considered a type of forward-backward filtering (e.g., `filtfilt` in Matlab or `scipy.signal.filtfilt` in Python). An advantage of (6) is that transient effects at the start and end of the finite-length signal are intrinsically avoided. This is because each row of the convolution matrices $\mathbf{P}$ and $\mathbf{Q}$ in (1) and (2) contains a full impulse response (not truncated). (This is akin to the `valid` option in the Matlab `conv` function or the Python `numpy.convolve` function.) Therefore, using expression (6), it is not necessary to specify internal filter states or perform symmetric extensions, which are usual ways to avoid transient end-point artifacts in forward-backward filtering.

The implementation of (6) requires the solution of a system of linear equations. Fast memory-efficient solvers for banded systems can be used for this purpose [28, Sect 2.4].

2.1. High-pass Filter

If H is a zero-phase low-pass filter, then $G = I - H$ is a zero-phase high-pass filter with transfer function

$$G(z) = \frac{\alpha P(z)P(1/z)}{Q(z)Q(1/z) + \alpha P(z)P(1/z)}. \tag{12}$$

Using (8) and (9), the filter matrices for G are given by

$$\begin{aligned} \mathbf{G} &= \mathbf{I} - \mathbf{H} && (13)\\ &= \mathbf{I} - \mathbf{A}^{-1}\mathbf{Q}^{\mathsf{T}}\mathbf{Q} && (14)\\ &= \mathbf{A}^{-1}(\mathbf{A} - \mathbf{Q}^{\mathsf{T}}\mathbf{Q}) && (15)\\ &= \alpha\mathbf{A}^{-1}\mathbf{P}^{\mathsf{T}}\mathbf{P} && (16) \end{aligned}$$

where we used (8) to simplify (15).

2.2. Butterworth Low-pass Filter

Some classical filters have transfer functions of the form (11). From (11), note that

$$\begin{aligned} P(z_0) = 0 &\implies H(z_0) = 1 && (17)\\ Q(z_0) = 0 &\implies H(z_0) = 0. && (18) \end{aligned}$$

In this paper, we set

$$\begin{aligned} P(z) &= (1 - z^{-1})^d && (19)\\ Q(z) &= (1 + z^{-1})^d && (20) \end{aligned}$$

for a positive integer d, leading to $P(1) = 0$ and $Q(-1) = 0$, i.e., $H(1) = 1$ and $H(-1) = 0$. Hence, the frequency response of the filter H has unity gain at $\omega = 0$ and a null at $\omega = \pi$. Furthermore, the frequency response is flat at these two points (its first few derivatives are zero, depending on d).

With $P(z)$ and $Q(z)$ given by (19) and (20), the filter H in (11) is a discrete-time Butterworth filter [27]. The frequency response of H is given by

$$H(\mathrm{e}^{\mathrm{j}\omega}) = \frac{\cos^{2d}(\omega/2)}{\cos^{2d}(\omega/2) + \alpha\sin^{2d}(\omega/2)}. \tag{21}$$

We can set α so the frequency response has a designated cut-off frequency ω_c. Setting $H(\mathrm{e}^{\mathrm{j}\omega_c}) = 0.5$ and solving for α yields

$$\alpha = 1/\tan^{2d}(\omega_c/2). \tag{22}$$

The zero-phase Butterworth filter, implemented for finite-length signals using (6), exactly preserves polynomial input signals up to degree $2d-1$ (with no transients at signal end-points). This is due to the flatness of $H(z)$ at $z = 1$. The Savitzky-Golay filter [30] also has this polynomial approximation property.

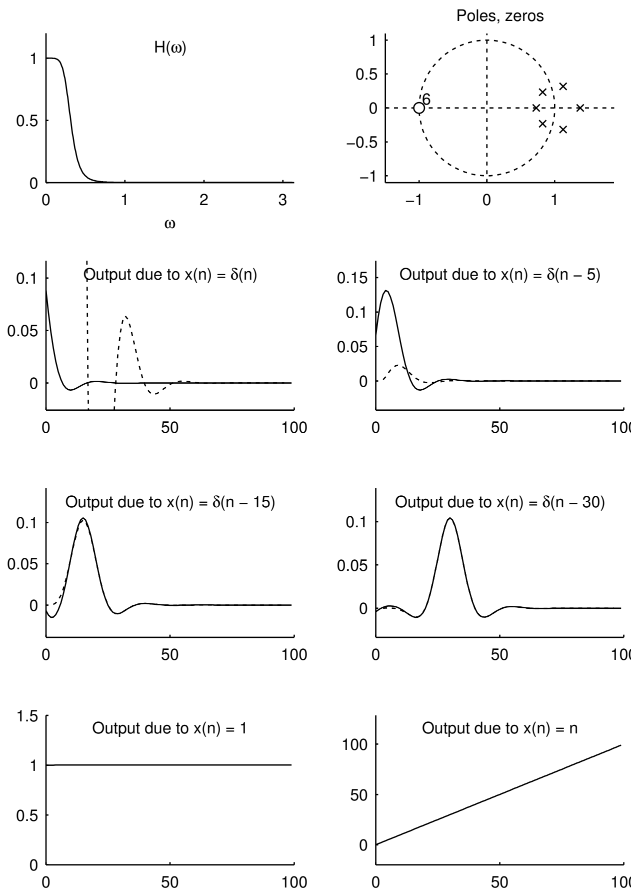

Fig. 1: Zero-phase Butterworth filter for finite-length data. (The dashed line is the old formulation which has end-point transients.)

With $P(z)$ and $Q(z)$ given by (19) and (20), the matrices $\mathbf{P}$ and $\mathbf{Q}$ consist of binomial coefficients. For example, when $d = 3$, the matrices are given by

$$\mathbf{Q} = \begin{bmatrix} 1 & 3 & 3 & 1 & & & \\ & 1 & 3 & 3 & 1 & & \\ & & \ddots & & & \ddots & \\ & & & 1 & 3 & 3 & 1 \end{bmatrix} \tag{23}$$

$$\mathbf{P} = \begin{bmatrix} -1 & 3 & -3 & 1 & & & \\ & -1 & 3 & -3 & 1 & & \\ & & \ddots & & & \ddots & \\ & & & -1 & 3 & -3 & 1 \end{bmatrix}. \tag{24}$$

These matrices are of size $(N-d) \times N$ where N denotes the length of the vector to which the matrices are applied.

Figure 1 illustrates the zero-phase Butterworth filter for $d = 3$ and $\omega_c = 0.1\pi$. To illustrate the absence of transient artifacts at end-points of a finite-length signal, the output signals due to six input signals are shown, where the signal length is $N = 100$. The output signals are computed using (6). The filter preserves polynomial input signals up to degree 5 (i.e., $2d - 1 = 5$). For example, Fig. 1 shows that the filter preserves input polynomial signals of degree 0 and 1 (i.e., constant and line). The figure also shows the original formulation (in dashed line-style) which exhibits unwanted transient artifacts at the signal end-points [31].

2.3. Factorization

We define $\mathbf{D}$ as the matrix representing K-order discrete-time difference, e.g., when $K = 2$ we have

$$\mathbf{D} = \begin{bmatrix} 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & \ddots & & \ddots \\ & & & 1 & -2 & 1 \end{bmatrix}. \tag{25}$$

The matrix $\mathbf{D}$ corresponds to the LTI transfer function

$$D(z) = (1 - z^{-1})^K. \tag{26}$$

The matrix $\mathbf{D}$ is of size $(N - K) \times N$.

If $K \leqslant d$, then $P(z)$ in (19) can be expressed as

$$\begin{aligned} P(z) &= (1 - z^{-1})^d & (27) \\ &= P_1(z)\, D(z) & (28) \end{aligned}$$

where we define

$$P_1(z) = (1 - z^{-1})^{d-K}. \tag{29}$$

Correspondingly, we write the matrix $\mathbf{P}$ as the product

$$\mathbf{P} = \mathbf{P}_1 \mathbf{D} \tag{30}$$

where $\mathbf{P}_1$ is the Toeplitz matrix corresponding to $P_1(z)$. The matrix $\mathbf{P}_1$ is of size $(N - d) \times (N - K)$. We will use relation (30) to simplify the cost function below.

3. SPARSITY-ASSISTED SIGNAL SMOOTHING

We assume the signal x to be estimated can be expressed as

$$x(n) = x_1(n) + x_2(n), \;\; n \in \mathbb{Z} \tag{31}$$

where signal x_1 has a sparse K-order derivative and x_2 is a low-frequency signal. We assume the signal x is corrupted by additive white Gaussian noise (AWGN),

$$y(n) = x(n) + w(n), \;\; n \in \mathbb{Z} \tag{32}$$

where w is the noise. We assume that if the sparse-derivative component x_1 were absent, then low-pass filtering would be sufficient to estimate x_2 (since x_2 is a low-frequency signal by assumption). Hence, if x_1 were known, we may estimate x_2 by subtracting x_1 from the noisy data y and low-pass filtering, i.e.,

$$\hat{x}_2 = \mathsf{LPF}\{y - \hat{x}_1\} \tag{33}$$

where LPF represents a zero-phase low-pass filter. So, we propose to estimate x as

$$\hat{x} = \hat{x}_1 + \mathsf{LPF}\{y - \hat{x}_1\} \tag{34}$$

where $\hat{x}_1$ is an estimate of x_1. Since the low-pass filter is linear, we can write

$$\begin{aligned} \hat{x} &= \hat{x}_1 + \mathsf{LPF}\{y\} - \mathsf{LPF}\{\hat{x}_1\} & (35) \\ &= \mathsf{LPF}\{y\} + \mathsf{HPF}\{\hat{x}_1\} & (36) \end{aligned}$$

where HPF represents the zero-phase high-pass filter $\mathsf{I} - \mathsf{LPF}$ where I is the identity operator. Using low-pass filter $\mathbf{H}$ in (9) and corresponding high-pass filter $\mathbf{G}$ in (16), we have

$$\hat{\mathbf{x}} = \mathbf{A}^{-1}\mathbf{Q}^{\mathsf{T}}\mathbf{Q}\mathbf{y} + \alpha\mathbf{A}^{-1}\mathbf{P}^{\mathsf{T}}\mathbf{P}\mathbf{x}_1 \tag{37}$$

where $\mathbf{x}_1$ is a signal with sparse K-order derivative, i.e., $\mathbf{D}\mathbf{x}_1$ is sparse. Let us use $P(z)$ and $G(z)$ with parameter d with $K \leqslant d$. Then $\mathbf{P} = \mathbf{P}_1\mathbf{D}$ from (30), so we write (37) as

$$\hat{\mathbf{x}} = \mathbf{A}^{-1}\mathbf{Q}^{\mathsf{T}}\mathbf{Q}\mathbf{y} + \alpha\mathbf{A}^{-1}\mathbf{P}^{\mathsf{T}}\mathbf{P}_1\mathbf{D}\mathbf{x}_1. \tag{38}$$

Since $\mathbf{D}\mathbf{x}_1$ is sparse, we write the signal model as

$$\hat{\mathbf{x}} = \mathbf{A}^{-1}\mathbf{Q}^{\mathsf{T}}\mathbf{Q}\mathbf{y} + \alpha\mathbf{A}^{-1}\mathbf{P}^{\mathsf{T}}\mathbf{P}_1\mathbf{u} \tag{39}$$

where $\mathbf{u}$ is sparse (and is to be determined/optimized).

Hence, a suitable cost function to determine $\mathbf{u}$ is

$$J(\mathbf{u}) = \frac{1}{2}\|\mathbf{y} - \mathbf{A}^{-1}\mathbf{Q}^{\mathsf{T}}\mathbf{Q}\mathbf{y} - \alpha\mathbf{A}^{-1}\mathbf{P}^{\mathsf{T}}\mathbf{P}_1\mathbf{u}\|_2^2 + \lambda\|\mathbf{u}\|_1$$

where the ℓ_1 norm is used to induce sparsity of $\mathbf{u}$. The quadratic data fidelity term corresponds to the additive white Gaussian noise assumption. Using (16), we have

$$\mathbf{y} - \mathbf{A}^{-1}\mathbf{Q}^{\mathsf{T}}\mathbf{Q}\mathbf{y} = \alpha\mathbf{A}^{-1}\mathbf{P}^{\mathsf{T}}\mathbf{P}\mathbf{y}, \tag{40}$$

i.e.,

$$y - \mathsf{LPF}\{y\} = \mathsf{HPF}\{y\}, \tag{41}$$

so the cost function to be minimized can be written as

$$J(\mathbf{u}) = \frac{\alpha^2}{2}\|\mathbf{A}^{-1}\mathbf{P}^{\mathsf{T}}\left(\mathbf{P}\mathbf{y} - \mathbf{P}_1\mathbf{u}\right)\|_2^2 + \lambda\|\mathbf{u}\|_1 \tag{42}$$

To clarify/summarize the matrix sizes: If $\mathbf{y} \in \mathbb{R}^N$, then

$$\mathbf{u} \in \mathbb{R}^{N-K}, \qquad \mathbf{P}_1 \in \mathbb{R}^{(N-d)\times(N-K)}, \tag{43}$$

$$\mathbf{A} \in \mathbb{R}^{N\times N}, \qquad \mathbf{P} \in \mathbb{R}^{(N-d)\times N}. \tag{44}$$

The minimization of the cost function J in (42) is the standard ℓ_1-norm sparse least squares problem arising in basis pursuit denoising, compressed sensing [6], etc. It can be solved by the iterative shrinkage/thresholding algorithm (ISTA) [10, 15] (an instance of forward-backward splitting [7, 8]), accelerated variants of ISTA (FISTA [2], FASTA [18], etc.), alternating direction method of multipliers (ADMM) [1, 17], iterative reweighted least-squares (IRLS) [19], etc. Therefore, we omit details about how to perform the minimization of the cost function J. We note only that all matrices $\mathbf{A}$, $\mathbf{P}$, and $\mathbf{P}_1$ are banded; thus, algorithms can be implemented with high computational efficiency, as described in [31].

A key point in the above formulation is that, even though we model $\mathbf{D}\mathbf{x}_1$ as sparse, the matrix $\mathbf{D}$ does not appear in the ℓ_1-norm penalty term of the cost function (42). Instead of penalizing $\mathbf{D}\mathbf{x}_1$, we penalize $\mathbf{u}$. This simplifies the problem and its algorithmic solution. This simplification is possible because $\mathbf{D}$ is a factor of $\mathbf{P}$ as related in (30), i.e., $D(z)$ is a factor of $P(z)$.

4. EXAMPLE

This example uses sparsity-assisted signal smoothing (SASS) to denoise the noisy ECG signal shown in Fig. 2(a). We simulate the ECG signal using ECGSYN [25] and add white Gaussian noise. The ECG signal exhibits abrupt changes in its slope. So, it seems reasonable to model the ECG signal as a low-frequency signal plus a signal with a sparse second-order derivative (i.e., with jump discontinuities in its first-order derivative). Hence, we set $K = 2$ in SASS.

For SASS, the low-pass filter H must also be specified. We set $d = 2$ which satisfies $K \leqslant d$ as required. We set the cut-off frequency to $\omega_c = 0.06\pi$ (i.e., $f_c = 0.03$ in normalized frequency units). This leads to $\alpha = 12524.52$ using (22). The parameters d and α define the zero-phase Butterworth filters H and G, i.e., the matrices $\mathbf{P}$, $\mathbf{Q}$, and $\mathbf{A}$. As Fig. 2(b) shows, this low-pass filter by itself substantially suppresses the noise and smooths the signal. But it severely distorts the sharp peaks of the ECG waveform.

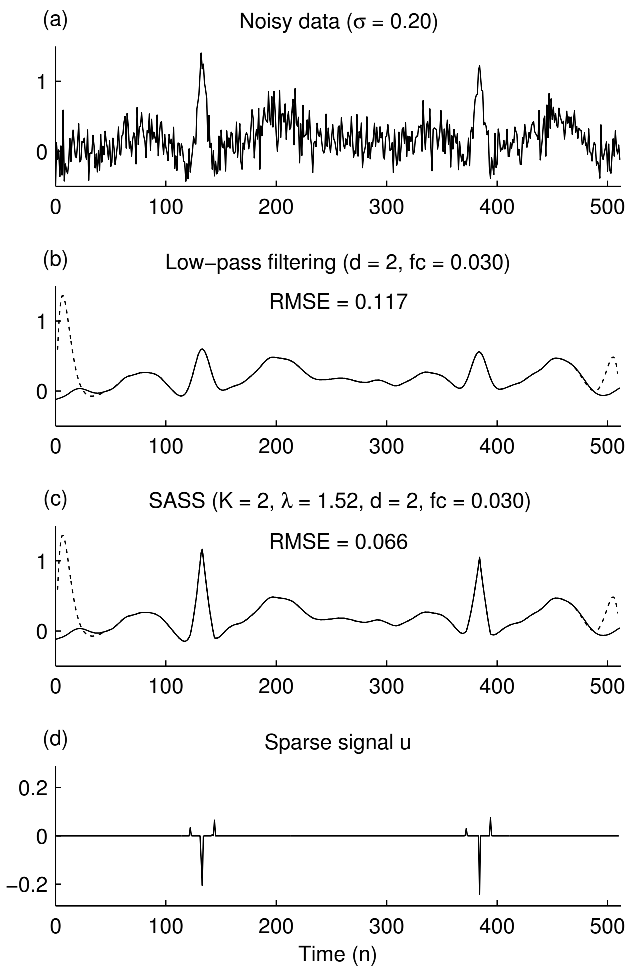

Fig. 2: Denoising of a simulated ECG signal. (The dashed line is the old formulation which has end-point transient artifacts.)

The output of SASS, shown in Fig. 2(c), accurately preserves the sharp peaks of the ECG waveform. Indeed, the first-order derivative of the SASS-denoised signal has jump discontinuities in several places. The sparse signal u, computed as part of the method, is shown in Fig. 2(d). It is the non-zero values of u that produce the jump discontinuities of the first-order derivative of the denoised signal. The denoised signal is given by (39) where u is obtained by minimizing the cost function (42). We have set the value of the parameter λ using the method suggested in [31]. As Fig. 2 shows, the proposed reformulation of SASS avoids the occurrence of unwanted end-point transient artifacts.

5. CONCLUSION

Sparsity-assisted signal smoothing (SASS) depends on a banded matrix formulation of recursive filtering of finite-length input signals. This paper presents a formulation that avoids the unwanted transient artifacts at signal end-points in the original formulation [31, 34].

6. REFERENCES

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

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

[3] K. Bredies, K. Kunisch, and T. Pock. Total generalized variation. *SIAM J. Imag. Sci.*, 3(3):492–526, 2010.

[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] E. J. Candes and M. B. Wakin. An introduction to compressive sampling. *IEEE Signal Processing Magazine*, 25(2):21–30, March 2008.

[7] P. L. Combettes and J.-C. Pesquet. Proximal thresholding algorithm for minimization over orthonormal bases. *SIAM J. Optim.*, 18(4):1351–1376, 2008.

[8] 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*, pages 185–212. Springer-Verlag, 2011.

[9] 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.

[10] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Commun. Pure Appl. Math*, 57(11):1413–1457, 2004.

[11] Y. Ding and I. W. Selesnick. Sparsity-based correction of exponential artifacts. *Signal Processing*, 120:236–248, March 2016.

[12] P. L. Dragotti and M. Vetterli. Wavelet footprints: theory, algorithms, and applications. *IEEE Trans. Signal Process.*, 51(5):1306–1323, May 2003.

[13] S. Durand and J. Froment. Reconstruction of wavelet coefficients using total variation minimization. *SIAM J. Sci. Comput.*, 24(5):1754–1767, 2003.

[14] 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.

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

[16] 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.

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

[18] T. Goldstein, C. Studer, and R. Baraniuk. A field guide to forward-backward splitting with a FASTA implementation. http://arxiv.org/abs/1411.3406, 2014.

[19] G. Harikumar and Y. Bresler. A new algorithm for computing sparse solutions to linear inverse problems. In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, volume 3, pages 1331–1334, May 1996.

[20] T.-C. Hsung, D. P. Lun, and W.-C. Siu. Denoising by singularity detection. *IEEE Trans. Signal Process.*, 47(11):3139–3144, 1999.

[21] Y. Hu and M. Jacob. Higher degree total variation (HDTV) regularization for image recovery. *IEEE Trans. Image Process.*, 21(5):2559–2571, May 2012.

[22] 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.

[23] 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.

[24] 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.

[25] P. E. McSharry, G. D. Clifford, L. Tarassenko, and L. A. Smith. A dynamical model for generating synthetic electrocardiogram signals. *IEEE Trans. Biomed. Eng.*, 50(3):289–294, March 2003.

[26] X. Ning, I. W. Selesnick, and L. Duval. Chromatogram baseline estimation and denoising using sparsity (BEADS). *Chemometrics and Intelligent Laboratory Systems*, 139:156–167, December 2014.

[27] T. W. Parks and C. S. Burrus. *Digital Filter Design*. John Wiley and Sons, 1987.

[28] 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.

[29] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.

[30] A. Savitzky and M. J. E. Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry*, 36(8):1627–1639, July 1964.

[31] I. W. Selesnick. Sparsity-assisted signal smoothing. In R. Balan et al., editors, *Excursions in Harmonic Analysis, Volume 4*, pages 149–176. Birkhäuser Basel, 2015.

[32] 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.

[33] I. W. Selesnick, H. L. Graber, Y. Ding, T. Zhang, and R. L. Barbour. Transient artifact reduction algorithm (TARA) based on sparse optimization. *IEEE Trans. Signal Process.*, 62(24):6596–6611, December 2014.

[34] 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.

[35] 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.