Abstract—This paper addresses the suppression of transient artifacts in signals, e.g., biomedical time series. To that end, we distinguish two types of artifact signals. We define ‘Type 1’ artifacts as spikes and sharp, brief waves that adhere to a baseline value of zero. We define ‘Type 2’ artifacts as comprising approximate step discontinuities. We model a Type 1 artifact as being sparse and having a sparse time-derivative, and a Type 2 artifact as having a sparse time-derivative. We model the observed time series as the sum of a low-pass signal (e.g., a background trend), an artifact signal of each type, and a white Gaussian stochastic process. To jointly estimate the components of the signal model, we formulate a sparse optimization problem and develop a rapidly converging, computationally efficient iterative algorithm denoted TARA (‘transient artifact reduction algorithm’). The effectiveness of the approach is illustrated using near infrared spectroscopic time-series data.

I. INTRODUCTION

This paper addresses the suppression of artifacts in measured signals, where the artifacts are of unknown shape but are known to be transient in form. We are motivated in particular by the problem of attenuating artifacts arising in biomedical time series, such as those acquired using near infrared spectroscopic (NIRS) imaging devices [3]. Our approach is based on a signal model intended to capture the primary characteristics of the artifacts, and on the subsequent formulation of an optimization problem. We model the measured discrete-time series, $y(n)$, as

$$y(n) = f(n) + x_1(n) + x_2(n) + w(n) \quad n \in \mathbb{Z},$$

where $f$ is a low-pass signal, $x_1$ are two distinct types of artifact signals, and $w$ is white Gaussian noise. Specifically, $f$ is low-pass in the sense that when $f$ is used as the input to an appropriately chosen high-pass filter, denoted by $H$, the output is approximately the all-zero signal; i.e., $Hf \approx 0$.

The ‘Type 1’ artifact signal, $x_1$, is intended to model spikes and sharp, brief waves, while the ‘Type 2’ artifact signal, $x_2$, is intended to model additive step discontinuities. For the purpose of flexibility and generality, we avoid defining the artifact signals in terms of precise rules or templates. Instead, we use the notion of sparsity to define them in a less regimented way that facilitates the formulation of an optimization-based approach:

1) We define the ‘Type 1’ artifact signal, $x_1$, as being sparse and having a sparse derivative (actually, a discrete-time approximation of the derivative, here and subsequently).

2) We define the ‘Type 2’ artifact signal, $x_2$, as having a sparse derivative. This type of artifact signal is composed of step discontinuities or approximate step discontinuities.

After the artifacts, $x_i$, are estimated, they are subtracted from the raw data to obtain a corrected time series.

To handle both types of artifacts simultaneously, in this work we develop an algorithm, denoted ‘Transient Artifact Reduction Algorithm’ (TARA). Complex artifacts often comprise both types; hence TARA performs joint optimization to maximize the effectiveness of the model to better reduce such artifacts. We devise TARA to have high computational efficiency and low memory requirements by constraining all matrices to be banded,\(^1\) which allows us to leverage fast solvers for banded systems [29, Sect 2.4]. Additionally, TARA does not require the user to specify auxiliary algorithmic parameters.

In addition to suppressing artifacts according to the model (1), we also consider the simplified model

$$y(n) = f(n) + x_1(n) + w(n), \quad n \in \mathbb{Z},$$

which contains artifacts of Type 1 only. We considered model (2) in previous work [34], where we used it to formulate what we called the ‘LPF/CSD’ problem. In this paper we present an improved algorithm for the LPF/CSD problem and provide an approach for setting the parameters. The new algorithm serves as the basis for the development of TARA, which suppresses artifacts according to model (1).

Although the suppression of Type 1 and Type 2 artifacts was addressed in our previous work [34], it was assumed that the measured time series is affected by the presence of either Type 1 or Type 2 artifacts, but not both. Two algorithms, one for each artifact type, are described in [34]. The respective algorithms were illustrated on time series acquired using a NIRS system [23], which frequently have artifacts of both types [15]. The algorithms were compared with the algorithm used by the NAP software application for NIRS artifact suppression [15].

\(^1\)A matrix is banded if its non-zero elements lie only on its main diagonal and a few adjacent upper and lower diagonals.
A. Related Work

Several approaches have been developed for the suppression of artifacts in biomedical time series [2], [9], [15], [20], [21], [25], [31], [37], [38]. Some methods, such as those based on independent component analysis (ICA) or adaptive filtering, require the availability of multiple channels or reference signals. However, if artifacts differ substantially among channels or if multiple channels are unavailable, then single-channel methods are needed [9], [21], [24]. Several methods for detecting and/or correcting motion artifacts in NIRS time series have been compared [9], [21], [24], [31], [32], leading to the conclusion that wavelet-based methods are more effective than other methods, especially for single-channel processing. Wavelets have also been shown effective for reducing ocular artifacts in EEG [2], [22]. Hence, we compare the sparse optimization and wavelet approaches below.

We remark that models (1) and (2) are prompted by the approach of morphological component analysis (MCA), in which all signal components are modeled as sparse with respect to distinct transforms [35]. However, the presence of the low-pass non-sparse component, \( f \), which differs from MCA, makes models (1) and (2) more realistic for biomedical time-series analysis. Moreover, modeling the low-pass component enhances the prospective sparsity of the remaining components, on which sparse optimization and MCA rely.

Rather than developing optimization schemes for general linear inverse problems, this work emphasizes algorithms that exploit the banded structure of one-dimensional linear time-invariant (LTI) operators to achieve high computational efficiency and fast convergence while avoiding additional algorithm parameters (e.g., step-size parameters), for the specific problems considered here. Yet, we note that general optimization algorithms, such as those based on proximity operators [8], [10], [12], [13], [28], [30], allow for consideration of non-smooth compound regularization problems more general than those considered here. Relevant surveys are also given in [4], [36].

B. Notation

Vectors and matrices are represented by lower- and upper-case bold (e.g., \( \mathbf{x} \) and \( \mathbf{H} \)), respectively. The \( n \)-th component of a vector \( \mathbf{x} \) is denoted \( [\mathbf{x}]_n \). Finite-length discrete-time signals are represented as vectors in \( \mathbb{R}^N \). The notation \( x \in [a, b] \) means \( a \leq x \leq b \).

II. Type 1 Artifacts (the LPF/CSD Problem)

In this section, we consider model (2), which contains artifacts of Type 1 only. The derivation of TARA in Sec. III for model (1) builds on the algorithm developed here. In previous work [34], an iterative algorithm based on the ‘alternating direction method of multipliers’ (ADMM) was derived for the ‘LPF/CSD’ problem. Here, we revisit the problem and present several improvements in comparison with [34].

1) A faster algorithm. In this work, we derive an algorithm based on majorization-minimization (MM). The new algorithm converges in fewer iterations in practice than the previous algorithm based on ADMM.

2) Fewer algorithm parameters. The algorithm of [34] required the user to specify a positive scalar, \( \mu \), analogous to a step-size parameter. A poor choice of \( \mu \) leads to slow convergence. The new MM algorithm does not require any parameters beyond those in the objective function.

3) A method to set regularization parameters, \( \lambda_0 \) and \( \lambda_1 \), based on the noise variance (which we assume is known).

4) A more general problem formulation. In this work, we allow the penalty functions to be non-convex, whereas [34] considered convex penalty functions only. With convex penalty functions, the amplitudes of artifacts tend to be underestimated (the estimates are biased toward zero).

5) A method to set non-convexity parameters. In the case where non-convex penalty functions are utilized, we present a method to set the non-convexity parameters. The previous work considered only convex penalty functions.

A. Problem formulation

We address the problem in the discrete-time setting. Signals are represented as vectors in \( \mathbb{R}^N \). We write model (2) as

\[
y = f + x + w, \quad y, x, w \in \mathbb{R}^N
\]

where the signal \( x \) is modeled as sparse and having a sparse derivative. The derivative is approximated using the discrete-time first-order difference operator, \( D \), defined by \( [D\mathbf{x}]_n = [\mathbf{x}]_{n+1} - [\mathbf{x}]_n \). The matrix \( D \) has the form

\[
D = \begin{bmatrix}
-1 & 1 & 0 & \cdots & \cdots & \cdots \\
-1 & -1 & 1 & \cdots & \cdots & \cdots \\
\vdots & \vdots & \ddots & \ddots & \ddots & \ddots \\
\vdots & \vdots & \cdots & -1 & 1 & 0 \\
\vdots & \vdots & \cdots & \cdots & \cdots & -1 & 1
\end{bmatrix}.
\]

In order to estimate \( x \) from \( y \), we propose to solve the optimization problem

\[
\hat{x} = \arg \min_x \left\{ F(x) = \frac{1}{2} \| \mathbf{H}(y - x) \|_2^2 + \lambda_0 \sum_n \phi_0([x]_n) + \lambda_1 \sum_n \phi_1([D\mathbf{x}]_n) \right\},
\]

where \( \lambda_1 > 0 \) are the regularization parameters. The low-pass signal is then estimated as

\[
\hat{f} = L(y - \hat{x}) = y - \hat{x} - \mathbf{H}(y - \hat{x}),
\]

where \( L \) denotes the low-pass filter defined as \( L = \mathbf{I} - \mathbf{H} \). The penalty functions, \( \phi_i : \mathbb{R} \rightarrow \mathbb{R} \), are chosen to promote sparsity. We refer to (5) as the LPF/CSD (low-pass-filtering/compound-sparse-denoising) problem [34].

When \( \phi_1 \) is the absolute-value function, \( \phi_1(x) = |x| \), for \( i = 1, 2 \), then problem (5) is the same as that considered in [34]. When, in addition, \( \mathbf{H} \) is the identity operator, (5) is the same as the ‘fused lasso signal approximator’ in Ref. [17].

The high-pass filter, \( \mathbf{H} \), is taken to be a zero-phase recursive discrete-time filter that we write as

\[
\mathbf{H} = \mathbf{B} \mathbf{A}^{-1},
\]

where \( \mathbf{A} \) and \( \mathbf{B} \) are banded Toeplitz matrices, as described in Sec. VI of [34]. We further suppose that \( \mathbf{B} \) admits the factorization

\[
\mathbf{B} = \mathbf{B}_1 \mathbf{D},
\]
where \( B_1 \) is banded and \( D \) is the above-noted first-order difference matrix. (See [34] for derivations of the factorizations in (7) and (8).) The fact that \( A \) and \( B \) are banded is important for the computational efficiency of the algorithm to be developed. We also assume that \( A, B_1, \) and \( D \) commute. As LTI filters, these operators are commutative for infinite-length discrete-time signals defined on \( \mathbb{Z} \). For finite-length signals, with which we deal in practice, the operators are approximately commutative, with the error being confined to the start and end of the signal, with a temporal extent that depends on the time-constant of the filter. We note that \( H \) was expressed as \( A^{-1}B \) in our earlier work [34]. Here we express \( H \) as \( BA^{-1} \), which the commutativity property permits, because the derivation of the computationally efficient MM algorithm in Sec. II-B relies on this ordering. We assume in this work that the signals of interest are sufficiently long that start and end transients are not problematic, in which case the commutativity assumption is justified.

The regularization parameters, \( \lambda_0 \) and \( \lambda_1 \), control the relative weight between the penalty terms. Their values should also be set according to the noise variance: higher noise calls for higher \( \lambda_1 \). As noted in [34], the regularization in problem (5) is an example of compound regularization [1], [6], wherein two or more regularizers are used to promote distinct properties of the signal to be recovered.

In (5), the functions \( \phi_i : \mathbb{R} \to \mathbb{R} \) are chosen to be sparsity-promoting penalty functions more general than the \( \ell_1 \) norm. Non-smooth penalties are given in lines 1a-3a of Table I; these are non-differentiable at zero. For the logarithmic (log) and arctangent (atan) penalties, the parameter \( a > 0 \) controls the extent to which the functions are non-convex. The log and atan penalties are strictly concave for \( a > 0 \). As \( a \to 0 \), the log and atan penalties approach the absolute value function. The atan penalty was derived to promote sparsity more strongly than the log penalty [33]. Additionally, it will be useful below to define \( \psi : \mathbb{R} \to \mathbb{R} \) as \( \psi(u) := u/\phi'(u) \).

The algorithms derived in Sec. II-B and Sec. III-A inevitably encounter ‘divide-by-zero’ errors when any of the penalties are non-differentiable at zero. To prevent this error condition, we introduce a small degree of smoothing so that the penalty functions become differentiable at \( u = 0 \); see lines 1b-3b of Table I. When the constant \( \epsilon \) is sufficiently small, the minimizer of the smoothed objective function function is negligibly different from the minimizer of the non-smoothed one. In practice, we use \( \epsilon = 10^{-8} \).

### B. Algorithm

We use the majorization-minimization (MM) procedure [16] to minimize the objective (cost) function, \( F : \mathbb{R}^N \to \mathbb{R} \), defined in (5). We use the MM procedure with a quadratic majorizer of the penalty function; each iteration of the MM procedure then requires the minimization of a quadratic function, which is performed by solving a system of linear equations. In this work, with suitable manipulations, the system matrix of the linear equations is banded. Hence, fast solvers for banded systems can be used to implement the algorithm with very high computational efficiency.

### Table I

<table>
<thead>
<tr>
<th>Penalty, ( \phi(u) )</th>
<th>( \psi(u) = u/\phi'(u) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a. (</td>
<td>u</td>
</tr>
<tr>
<td>2a. ( \frac{1}{a} \log(1 + a</td>
<td>u</td>
</tr>
<tr>
<td>3a. ( \frac{2}{a\sqrt{3}} \left( \tan^{-1} \left( \frac{1 + 2a</td>
<td>u</td>
</tr>
<tr>
<td>1b. ( \sqrt{u^2 + \epsilon} )</td>
<td>( \sqrt{u^2 + \epsilon} )</td>
</tr>
<tr>
<td>2b. ( \frac{1}{a} \log \left( 1 + a\sqrt{u^2 + \epsilon} \right) )</td>
<td>( \sqrt{u^2 + \epsilon} (1 + a \sqrt{u^2 + \epsilon}) )</td>
</tr>
<tr>
<td>3b. ( \frac{2}{a \sqrt{3}} \left( \tan^{-1} \left( \frac{1 + 2a(\sqrt{u^2 + \epsilon})}{\sqrt{3}} \right) - \frac{\pi}{2} \right) )</td>
<td>( \sqrt{u^2 + \epsilon} \left( 1 + a \sqrt{u^2 + \epsilon} + a^2(u^2 + \epsilon) \right) )</td>
</tr>
</tbody>
</table>

Note: \( u \in \mathbb{R}, \ a > 0, \ \epsilon > 0 \). 1b-3b are smoothed penalties.

Under suitable restrictions on \( \phi \) (symmetric, continuously differentiable, etc.), which are satisfied by the penalty functions in lines 1b-3b of Table I, a majorizer of \( \phi \) is given by

\[
g(u, v; \phi) := \frac{\phi'(v)}{2v} u^2 + \phi(v) - \frac{v}{2} \phi'(v). \tag{9}\]

That is,

\[
g(u, v; \phi) \geq \phi(u), \quad \text{for all } u, v \in \mathbb{R} \tag{10}\]

\[
g(v, v; \phi) = \phi(0), \quad \text{for all } v \in \mathbb{R}. \tag{11}\]

Note that \( g(u, v; \phi) \) is quadratic in \( u \). The majorizer \( g \) can be used, in turn, to obtain a majorizer of \( F \) in (5).

If \( u \) and \( v \) are vectors, then

\[
\sum_n g([u]_n, [v]_n; \phi_0) \geq \sum_n \phi_0([u]_n), \tag{12}\]

with equality if \( u = v \). Using (9), we write the left-hand-side of (12) compactly as

\[
\sum_n g([u]_n, [v]_n; \phi_0) = \frac{1}{2} u^T [\Lambda(v; \phi_0)] u + c_0, \tag{13}\]

where \( [\Lambda(v; \phi)] \) is defined as the diagonal matrix

\[
[\Lambda(v; \phi)]_{n,n} := \frac{\phi'([v]_n)}{[v]_n} = 1/\psi([v]_n) \tag{14}\]

and \( c_0 \) does not depend on \( u \). Similarly,

\[
\sum_n g([Du]_n, [Dv]_n; \phi_1) \geq \sum_n \phi_1([Du]_n), \tag{15}\]

with equality if \( u = v \). The left-hand-side can be written as

\[
\sum_n g([Du]_n, [Dv]_n; \phi_1) = \frac{1}{2} u^T D^T [\Lambda(Dv; \phi_1)] Du + c_1, \tag{16}\]

where \( \Lambda \) is defined as in (14) and \( c_1 \) does not depend on \( u \). Therefore, using (12) and (15), a majorizer of \( F \) in (5) is given by \( G : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R} \), defined by

\[
G(x, v) = \frac{1}{2} \|BA^{-1}(v - x)\|^2 + \frac{\lambda_0}{2} x^T [\Lambda(v; \phi_0)] x + \frac{\lambda_1}{2} x^T D^T [\Lambda(Dv; \phi_1)] Dx + c, \tag{16}\]
where \( c \) does not depend on \( x \). The MM update equation, wherein the majorizer is minimized, is

\[
x^{(i+1)} = \arg \min_x G(x, x^{(i)}),
\]

where \( i \) is the iteration index. The update equation (17) leads to

\[
x^{(i+1)} = \left( A^{-T}B^TBA^{-1} + A^{(i)}_0 + D^T\Lambda^{(i)}_1D \right)^{-1} \times A^{-T}B^TB^{-1}y,
\]

where \( \Lambda^{(i)}_0 \) and \( \Lambda^{(i)}_1 \) are the diagonal matrices

\[
\Lambda^{(i)}_0 := \lambda_0 \lambda_0(x^{(i)}; \phi_0), \quad \Lambda^{(i)}_1 := \lambda_1 \lambda_1(Dx^{(i)}; \phi_1).
\]

Specifically,

\[
[A^{(i)}_0]_{n,n} := \frac{\phi_0([x^{(i)}]_n)}{[x^{(i)}]_n} = \frac{\lambda_0 \psi_0([x^{(i)}]_n)}{\lambda_0 \psi_0([x^{(i)}]_n)}
\]

\[
[A^{(i)}_1]_{n,n} := \frac{\phi_1([Dx^{(i)}]_n)}{[Dx^{(i)}]_n} = \frac{\lambda_1 \psi_1([Dx^{(i)}]_n)}{\lambda_1 \psi_1([Dx^{(i)}]_n)}.
\]

Note that the system matrix in (18) (i.e., inside the parentheses), is not banded, because \( A^{-1} \) is not banded. Hence, fast solvers for banded systems cannot be used directly with (18). To utilize fast solvers, we write

\[
(A^{-T}B^TBA^{-1} + A^{(i)}_0 + D^T\Lambda^{(i)}_1D)^{-1} = A \left( B^TBA + A^{-1}(A^{(i)}_0 + D^T\Lambda^{(i)}_1D)A \right)^{-1} A^T
\]

where the matrix within the inverse is banded. Then, using (22) in (18), we obtain the update equation

\[
x^{(i+1)} = A \left( B^TBA + A^{-1}(A^{(i)}_0 + D^T\Lambda^{(i)}_1D)A \right)^{-1} B^TBA^{-1}y.
\]

The algorithm, summarized in Table II, can be implemented efficiently using fast solvers for banded systems because the matrix \( Q \) in line 5 is banded. The penalty functions \( \phi_i \) arise only in lines 3 and 4 of the algorithm (Table II) and their role is encapsulated by \( \psi_i \).

We note that, as the algorithm progresses, many elements of \([x]_n \) and \([Dx]_n \) generally go to zero when \( \phi_0 \) and \( \phi_1 \) are chosen to be any of the non-smooth penalties in lines 1a-3a of Table I, or other non-smooth sparsity-promoting penalties. Note also that, for such penalties, \( \psi_i(u) \to 0 \) as \( u \to 0 \). Therefore, as the algorithm converges to a sparse vector \( x \), elements of the \( \Lambda \) matrices go to infinity, and consequently the system of linear equations (line 6 in Table II) becomes ill-conditioned. For this reason, we use the smoothed versions of the penalty functions with a small value of \( \varepsilon \) to avoid ‘divide-by-zero’ errors. In practice, we set \( \varepsilon = 10^{-8} \). We have found that slightly smoothing the penalty term has a negligible effect on the minimizer \( \hat{x} \), as validated by the comparison of ADMM and MM algorithms in Fig. 4 below.

### C. Regularization parameters

To use the LPF/CSD formulation to estimate transient artifacts, the \( \lambda_0 \) and \( \lambda_1 \) regularization parameters must be specified. The process of specifying appropriate values is governed by two considerations that have a substantial effect on the estimated artifact signal \( \hat{x} \). First, the values of both parameters nominally should be proportional to \( \sigma \), where \( \sigma^2 \) is the noise variance, which we assume is known. In addition, we aim to set \( \lambda_1 \) so that \( \hat{x} \) contains, with high probability, signal behavior that is due solely to the artifacts present in the data. This is because we seek to correct the observed data \( y \) by subtracting \( \hat{x} \) from it. If \( \hat{x} \) exhibits behavior that is not due to artifacts, then subtracting it from \( y \) generally leads to distortion of the signal of interest (e.g., oscillations of biomedical origin). Accordingly, the artifact characteristics of the data is the second factor that governs the selection of parameter values.

We begin by considering how to set \( \lambda_1 \) according to the noise variance. First, note that if the \( \lambda_1 \) are sufficiently large, then \( \hat{x} \), the minimizer of \( F \), will be the all-zero vector, \( \hat{x} = 0 \). Second, note that if the \( \lambda_1 \) are near zero, then \( \hat{x} \) will approximate the noisy data, \( \hat{x} \approx y \). We seek to set the \( \lambda_1 \) large enough so that \( \hat{x} \) comprises solely artifact-related signal behavior, but not so large that \( \hat{x} \) is overly attenuated toward zero. Suppose that some realization of the data consists purely of noise, i.e., \( y = w \sim N(0, \sigma^2I) \). In this case, \( \hat{x} \) should be the all-zero vector, \( \hat{x} = 0 \), at least with high probability. Hence \( \lambda_1 \) should be chosen sufficiently large so that when \( y \) consists purely of noise, we obtain \( \hat{x} \approx 0 \) with high probability. That is, a suitable pair, \( (\lambda_0, \lambda_1) \), is one for which we can write

\[
y \sim N(0, \sigma^2I) \implies \hat{x} \approx 0 \quad \text{with high probability},
\]

where \( \hat{x} \) is obtained by solving (5). The suitable \( (\lambda_0, \lambda_1) \) thus depends on the noise standard deviation, \( \sigma \). To find such \( (\lambda_0, \lambda_1) \) analytically is difficult due to the compound regularization. To tackle this problem, we consider two special cases where selection of \( \lambda_1 \) that satisfy (23) is relatively straightforward.

In the first special case, we set \( \lambda_1 \) to zero and seek a value for \( \lambda_0 \) that satisfies (23). We denote this value \( \lambda_0^p \); i.e., \( (\lambda_0^p, 0) \) is a pair satisfying (23). Likewise, in the second special case, we set \( \lambda_0 \) to zero and seek a value for \( \lambda_1 \) that satisfies (23). We denote this value \( \lambda_1^p \); i.e., \( (0, \lambda_1^p) \) is a pair satisfying (23). Then for the general case we interpolate between these two pairs to obtain \( (\lambda_0, \lambda_1) \) approximately satisfying (23).

---

**TABLE II**

**MM algorithm solving the LPF/CSD problem (5)**

<table>
<thead>
<tr>
<th>Input: ( y \in \mathbb{R}^N, \lambda_i &gt; 0, \phi_i, A, B )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. ( y = B^TBA^{-1}y )</td>
</tr>
<tr>
<td>2. ( x = y ) (initialization)</td>
</tr>
<tr>
<td>Repeat</td>
</tr>
<tr>
<td>3. ([A_0]_{n,n} = \lambda_0 / \psi_0([x]_n)) (( A_0 ) is diagonal)</td>
</tr>
<tr>
<td>4. ([A_1]_{n,n} = \lambda_1 / \psi_1([Dx]_n)) (( A_1 ) is diagonal)</td>
</tr>
<tr>
<td>5. ( Q = B^TBA + A^{-1}(A_0 + D^T\Lambda_1D)A ) (( Q ) is banded)</td>
</tr>
<tr>
<td>6. ( x = QA^{-1}y ) (MM update)</td>
</tr>
<tr>
<td>Until convergence</td>
</tr>
</tbody>
</table>

**Output: \( x \)**
In order to find $\lambda_0^*$ and $\lambda_1^*$, we work with two special cases of (5). We define the objective function, $F_0 : \mathbb{R}^N \to \mathbb{R}$, as
\[
F_0(x) = \frac{1}{2} \|H(y - x)\|^2 + \lambda_0 \sum_n \phi_0([x]_n) \tag{24}
\]
and the objective function, $F_1 : \mathbb{R}^N \to \mathbb{R}$, as
\[
F_1(x) = \frac{1}{2} \|H(y - x)\|^2 + \lambda_1 \sum_n \phi_1([Dx]_n). \tag{25}
\]

The functions $F_0$ and $F_1$ correspond to $\lambda_1 = 0$ and $\lambda_0 = 0$ in (5), respectively. Unlike $F$, the $F_1$ do not involve compound regularization, which simplifies the analysis necessary to set $\lambda_i$. We denote the minimizers of $F_0$ and $F_1$ as:
\[
x_{F_0}^* = \arg \min_x F_0(x), \quad x_{F_1}^* = \arg \min_x F_1(x). \tag{26}
\]

To find $\lambda_0^*$ such that $(\lambda_0^*, 0)$ satisfies (23), we equivalently find $\lambda_0^*$ such that
\[
y \sim N(0, \sigma^2 I) \implies x_{F_0}^* \approx 0 \text{ with high probability.} \tag{27}
\]

That is, we seek to set $\lambda_0$ in (24) so that $x_{F_0}^*$ is relatively noise-free with high probability. The value for $\lambda_0^*$ accomplishing this is derived using optimality conditions from convex analysis. This general approach has been described by Fuchs [18] for the purpose of setting the false alarm rate in a target detection application. To find $\lambda_1^*$ such that $(0, \lambda_1^*)$ satisfies (23), we will proceed in a similar manner: the presence of $D$ in the penalty of $F$ needs to be taken into account.

**Obtaining $\lambda_0^*$**. We assume here that $\phi_0$ is one of the non-smooth penalty functions in Table I (lines 1a–3a). We will use a result from the theory of convex functions [5]: if a function $f : \mathbb{R}^N \to \mathbb{R}$ is convex, then $x$ is a minimizer of $f$ if and only if $0 \in \partial f(x)$, where $\partial f$ is the subgradient of $f$.

As shown in [33], if $\phi_0$ is chosen such that $F_0$ in (24) is convex, then $x$ minimizes $F_0$ if and only if, for all $n$,
\[
[H^T H(y - x)]_n = 0 \quad \text{if } \quad \lambda_0 \phi_0([x]_n), \quad [x]_n \neq 0 \\
\phantom{=} \in [-\lambda_0, \lambda_0], \quad [x]_n = 0. \tag{28}
\]

When $x = x_{F_0}^*$ is the all-zero vector, we have
\[
x_{F_0}^* = 0 \implies [H^T H y]_n \in [-\lambda_0, \lambda_0], \quad \forall n, \tag{29}
\]
and the right-hand-side of (27) can be written as $[H^T H y]_n \in [-\lambda_0, \lambda_0], \quad \forall n$ with high probability. Hence, $\lambda_0^*$ should be chosen so that
\[
\lambda_0^* \geq \|H^T H y\|_n, \quad \forall n \text{ with high probability}, \tag{30}
\]

where $y \sim N(0, \sigma^2 I)$. Note that $H^T H$ represents an LTI system. Then $H^T H y$ is a stationary stochastic process and $[H^T H y]_n \sim N(0, \sigma_0^2)$, where $\sigma_0$ is given by
\[
\sigma_0 := \text{std}([H^T H y]_n) = \|p_0\|_2 \sigma \tag{31}
\]

and $p_0$ is the impulse response of the LTI filter $H^T H$. That is, $p_0 = h \ast h^r$, where $h$ represents the impulse response of the LTI filter $H := BA^{-1}$, and $h^r$ is the time-reversed version of $h$ (i.e., $h^r(-n) = h(n)$).

So, (30) can be expressed as
\[
\lambda_0^* \geq |v|, \quad \forall n \text{ with high probability, where } v \sim N(0, \sigma_0^2). \tag{32}
\]

A nominal value of $\lambda_0^*$ is given by the ‘three-sigma’ rule,
\[
\lambda_0^* = 3 \sigma_0 = 3 \|p_0\|_2 \sigma. \tag{33}
\]

Using the $\lambda_0^*$ value given by (31), (30) is satisfied with a probability above 99%.

**Obtaining $\lambda_1^*$**. From (7) and (8), note that $H = B_1 A^{-1}$. Using commutativity, we have $H = B_1 A^{-1} D$. Hence constant-valued signals are in the null space of both $H$ and $D$; i.e., the signal $[x]_n = c$ is annihilated by both operators. Therefore, if $x_2$ is defined as $[x_2]_n = [x]_n + c$, then $F_1(x_2) = F_1(x)$; i.e., the value of the objective function $F_1$ is unaffected by a shift in the baseline of $x$. Then the signal minimizing $F_1$ in (25) is unique only up to an additive constant. This issue is addressed by defining a change of variables, namely $u = D x$, which facilitates the derivation of $\lambda_1^*$.

We define $H_1 = B_1 A^{-1}$. Then, $H_1 D = H$, and we can write
\[
Hx = H_1 u, \quad u = Dx. \tag{34}
\]

Hence, minimizing (25) is equivalent to the problem
\[
u_{\tilde{F}_1}^* = \arg \min_u \left\{ \tilde{F}_1(u) = \frac{1}{2} \|Hy - H_1 u\|^2 + \lambda_1 \sum_n \phi_1([u]_n) \right\}. \tag{35}
\]

Accordingly, to find $\lambda_1^*$ such that $(\lambda_1^*, 0)$ satisfies (23), we equivalently find $\lambda_1^*$ such that
\[
y \sim N(0, \sigma^2 I) \implies u_{\tilde{F}_1}^* \approx 0 \text{ with high probability.} \tag{36}
\]

As above, if $\phi_1$ is chosen such that $\tilde{F}_1$ is convex, then $u$ minimizes $\tilde{F}_1$ if and only if, for all $n$,
\[
[H_1^T (Hy - H_1 u)]_n = \lambda_1 \phi_1([u]_n), \quad [u]_n \neq 0 \\
\phantom{=} \in [-\lambda_1, \lambda_1], \quad [u]_n = 0. \tag{37}
\]

Proceeding as above, we obtain a nominal value for $\lambda_1^*$ of
\[
\lambda_1^* = 3 \sigma_1 = 3 \|p_1\|_2 \sigma, \tag{38}
\]

where $\sigma_1 := \text{std}([H_1^T H y]_n) = \|p_1\|_2 \sigma$ and $p_1$ is the impulse response of the LTI filter $H_1^T H$. That is, $p_1 = h \ast h^r$, where $h_1$ represents the impulse response of the LTI filter $H_1 := B_1 A^{-1}$, and $h_1^r$ is the time-reversed version of $h_1$.

**Setting $(\lambda_0, \lambda_1)$**. The parameter pair $(\lambda_0^*, 0)$ is appropriate for an artifact signal that is known to be sparse, i.e., departing only briefly from a baseline value of zero. In this case, the artifact signal can be modeled as consisting of pure impulses, i.e., isolated spikes of large deviation from baseline, such as ‘salt and pepper’ noise. This is because, when $\lambda_1 = 0$, the objective function imposes no continuity among the non-zero values of the artifact signal. On the other hand, the parameter pair $(0, \lambda_1^*)$ is suitable when it is known that the derivative of the artifact signal is sparse, i.e., the artifact signal consists of step discontinuities.

Real artifacts are generally not so easily classified as spikes or as additive step discontinuities. Therefore, $\lambda_0$ and $\lambda_1$ should be tuned according to the behavior of the artifacts in the data.
Although the values $\lambda_0^*$ and $\lambda_1^*$ in (31) and (35) are ideally suited for two special cases only, they provide anchors for the selection of $(\lambda_0, \lambda_1)$. We set

$$\lambda_0, \lambda_1 = (\theta \lambda_0^*, (1-\theta) \lambda_1^*), \quad 0 \leq \theta \leq 1,$$  \hfill (36) 

which restricts $(\lambda_0, \lambda_1)$ to a line segment in the plane, reducing the two degrees of freedom to one. As one of $(\lambda_0, \lambda_1)$ is reduced, the other increases. Reducing one parameter without increasing the other would lead to a total reduction in the regularization, leading to potential noise contamination of $\hat{x}$. Thereby, the interpolation (36) approximately satisfies (23); i.e., it takes into account the noise variance so that the estimated artifact signal $\hat{x}$ is largely noise-free with high probability. In this approach, one of the two degrees of freedom in the LPF/CSD problem (5) is set according to the noise variance, and the other is used to tune the algorithm to the data.

D. Noise model deviation

Real time series are likely to deviate from the idealized model (2) on which LPF/CSD is based. The mid- and high-frequency spectral content of the data may comprise a mixture of biologically relevant signals, rather than white noise. In such cases, there is no well-defined noise standard deviation to use in formulas (31) and (35). However, the approach can still be utilized by using a ‘pseudo-noise sigma’ that serves as a substitute. The pseudo-sigma parameter then leads to values for $\lambda_0^*$ and $\lambda_1^*$. The pseudo-σ parameter can be tuned using a representative data set. Then $\theta \in [0, 1]$ should be tuned such that $\hat{x}$ captures the transient artifacts most effectively. In this way, the problem formulation (2) is parameterized in terms of $(\sigma, \theta)$ instead of $(\lambda_0, \lambda_1)$. We have found this a more convenient parameterization for setting parameter values.

E. Setting the non-convexity parameters

The use of non-convex penalties in (5) can be advantageous because, in comparison with convex penalties, they generally produce estimates that are less biased toward zero; i.e., the amplitudes of the estimated transients are less attenuated than those produced by convex penalties [14]. However, when using non-convex penalties, optimization algorithms may get trapped in sub-optimal local minima. Hence, non-convex penalties should be specified with care. One approach to avoid the issue of entrapment in local minima is to specify non-convex penalties such that the total objective function, $F$, is convex [7, 26, 27, 33]. Then the total objective function, owing to its convexity, does not posses sub-optimal local minima and a global optimal solution can be reliably found. The design of non-convex penalties according to this principle is formulated as a semidefinite program (SDP) in [33]. Here, we make simplifying assumptions to avoid the high computational cost of SDP.

When we use the logarithmic or arctangent penalty functions, which are non-convex, we need to set the non-convexity parameter, $a$, for each of $\phi_0$ and $\phi_1$. We denote the respective values by $a_0$ and $a_1$, and write the penalties as $\phi_0(u, a_0)$ and $\phi_1(u, a_1)$ to emphasize the dependence of the penalties on $a_i$.

To derive a heuristic for setting the non-convexity parameters, we assume that the sparse vectors $x_0^{\text{opt}}$ and $a_0^{\text{opt}}$ contain only a single non-zero entry. While this assumption is not satisfied in practice, with it we obtain values of $a_i$ for which $F$ is definitely non-convex. Using corollary 1 of Ref [33], this assumption leads to upper bounds on $a_0$ and $a_1$ of $\|h_2^i\|^2/\lambda_0$ and $\|h_1^i\|^2/\lambda_1$, respectively, where $h$ and $h_1$ represent the impulse responses of the systems $H := BA^{-1}$ and $H_1 := B_1A^{-1}$, respectively. Because the assumption is idealized, the upper bounds are too high in general (i.e., they do not guarantee convexity of $F$). Therefore, in the examples below, we halve these values, i.e., we set

$$a_0 = 0.5 \frac{\|h_2^i\|^2}{\lambda_0}, \quad a_1 = 0.5 \frac{\|h_1^i\|^2}{\lambda_1}. \hfill (37)$$

In the non-convex case, we initialize the algorithm with the $\ell_1$-norm solution to reduce the likelihood of the algorithm becoming trapped in a poor local minimizer. Due to the convexity of the $\ell_1$-norm, the initialization does not matter.

We also note that it was assumed in the derivation of (31) and (35) that the total objective function, $F$, is convex. Hence, suitably constraining the penalties so that $F$ is at least approximately convex is further advantageous, as it approximately justifies the use of (31) and (35) in setting $\lambda_i$.

F. LPF/CSD Example 1

This example shows LPF/CSD processing as applied to the simulated signal illustrated in Fig. 1(a). This signal consists of two low-frequency sinusoids, several additive step-transients, and additive white Gaussian noise ($\sigma = 0.5$).

In this example, we compute the solution to the LPF/CSD problem using both the $\ell_1$-norm and atan penalties. For the filter, we use the second-order zero-phase Butterworth filter described in [34]. We use (31) and (35) to obtain $\lambda_0^*$ and $\lambda_1^*$, and (37) to specify the non-convexity parameters for the atan penalty functions. We run the algorithm in Table II for 50 iterations.

With $(\lambda_0, \lambda_1) = (\lambda_0^*, 0)$ and the atan penalty, we obtain the estimated transient signal $\hat{x}$ (i.e., $\hat{x} = x_0^{\text{opt}}$) shown in Fig. 2(a), which can be seen to deviate infrequently from the baseline value of zero. With $(\lambda_0, \lambda_1) = (0, \lambda_1^*)$ we obtain the estimated signal $\hat{x}$ (i.e., $\hat{x} = x_1^{\text{opt}}$) shown in Fig. 2(b), which does not follow a baseline of zero, but is approximately piecewise constant (i.e., has a sparse derivative). To obtain an estimated transient signal, $\hat{x}$, which both largely adheres to a baseline of zero and is piecewise constant, we set $(\lambda_0, \lambda_1)$ according to (36), with $\theta$ manually tuned to a value of 0.3. The result, shown in Fig. 2(c), is reasonably sparse, has a sparse derivative, and is not contaminated by the additive white Gaussian noise of the data. Reducing both $\lambda_0$ and $\lambda_1$ leads to a noisy under-regularized artifact signal. Increasing both $\lambda_0$ and $\lambda_1$ leads to an over-regularized artifact signal where the transient pulses are attenuated in amplitude. The interpolation given by (36) provides a trade-off between these two cases while keeping the total regularization at an appropriate level. Note that the interpolation (36) is in the domain of the regularization parameters, not the estimated signal $\hat{x}$ itself.

With $(\lambda_0, \lambda_1)$ as obtained in Fig. 2(c), the lowpass signal $\tilde{f}$ is obtained using (6). The total signal, $\tilde{f} + \hat{x}$, is shown
in Fig. 1(b) and (c) for the $\ell_1$-norm and atan penalties, respectively. It can be seen that the non-convex atan penalty estimates the amplitude of the transients more accurately than the $\ell_1$-norm penalty.

**G. LPF/CSD Example 2**

This example shows artifact reduction using LPF/CSD as applied to a near infrared spectroscopic (NIRS) time series. The NIRS neuroimaging data in this and later examples was acquired at a rate of 6.25 samples/second. Subjects were seated in a fixed chair (no wheels or reclining seat back, etc.), but were otherwise unrestrained. Participants were asked to remain as still and quiet as possible (no talking, etc.).

The data shown in Fig. 3(a) were acquired using a pair of optodes (one source and one detector) on the subject’s forehead near the left eye. As such, it is susceptible to artifacts due to eye blinks (in addition to other artifacts ordinarily present). Figure 3(a) shows transient artifacts of variable amplitude, width, and shape. The time series has a length of 1900 samples, of which 900 samples are shown to better reveal the signal details. The same time series was used in [34].

To apply LPF/CSD processing, we used a second-order zero-phase Butterworth filter with a cut-off frequency of 0.05 cycles/sample, and the atan penalty with $\lambda_0^*$ and $\lambda_1^*$ set as described in Sec. II-C, with a pseudo-noise standard deviation of $\sigma = 0.05$. The signals corresponding to ($\lambda_0^*$, 0) and (0, $\lambda_1^*$) are shown in Figs. 3(b) and (c), respectively. We then set $\lambda_0$ and $\lambda_1$ using (36) with $\theta$ manually tuned to 0.05, which produces the signal, $\hat{x}$, shown in Fig. 3(d). Only a small value of $\theta$ is needed to obtain a signal that adheres to a baseline value of zero. The result provides an apparently accurate estimate of the transient artifacts. The corrected time series, obtained by subtracting the estimated artifact signal $\hat{x}$ from the original data, is shown in Fig. 3(e). Compared with the result of [34], which used the $\ell_1$-norm penalty exclusively, the artifacts appear more accurately estimated here, due to the use of the atan penalty in place of the $\ell_1$ norm. The MM algorithm was run for 50 iterations, with a run time of about 80 milliseconds.

As noted above, if the LPF/CSD problem (5) is used with the the $\ell_1$-norm penalty, then ‘Algorithm 2’ in [34] (derived using ADMM) can be used instead of the algorithm in Table II (derived using MM). However, the ADMM algorithm has the disadvantage that it exhibits slower initial convergence than the MM algorithm, and it requires the user to specify a parameter, $\mu$, which, if improperly specified, can further reduce the initial convergence rate. To illustrate the comparative convergence behavior of the ADMM and MM algorithms, Fig. 4 shows the cost-function history for both. For the ADMM algorithm, the value of $\mu$ was optimized to give the smallest cost function value at iteration 50. We denote this optimal value by $\mu^\ast$. Increasing or decreasing $\mu$ adversely affects the behavior of the algorithm (with respect to a 50-iteration run). To illustrate this, we run the ADMM algorithm with $\mu$ set to $2\mu^\ast$ and $\mu^\ast/2$, in addition to $\mu^\ast$. Figure 4 shows that $2\mu^\ast$ has better
algorithm, which minimizes the non-smooth cost function, we evaluate the non-smooth cost function for both algorithms in Fig. 4. As the figure shows, even though the MM algorithm minimizes the smooth cost function, it reduces the non-smooth cost function faster than the ADMM algorithm over the first 50 iterations. (Eventually, the ADMM cost will cross below that of the MM cost, but at that point the solutions are negligibly different.) Hence, the slight smoothing of the penalty has an insignificant adverse impact in practice.

III. TRANSIENT ARTIFACT REDUCTION ALGORITHM

We address problem (1) in the discrete-time setting, and write the model as

$$y = f + x_1 + x_2 + w, \quad y, x_1, x_2, w \in \mathbb{R}^N$$  \hspace{1cm} (38)

where $f$ is a low-pass discrete-time signal, $x_1$ and $x_2$ comprise Type 1 and Type 2 artifacts, respectively, and $w$ is additive white Gaussian noise. That is, $x_1$, $Dx_1$, and $Dx_2$ are modeled as sparse. Similar to Sec. II, the discrete-time formulation leads to the optimization problem

$$\{\hat{x}_1, \hat{x}_2\} = \arg \min_{x_1, x_2}\left\{\frac{1}{2}\|H(y - x_1 - x_2)\|_2^2 + \lambda_0 \sum_n \phi_0([x_1]_n) + \lambda_1 \sum_n \phi_1([Dx_1]_n) + \lambda_2 \sum_n \phi_2([Dx_2]_n)\right\},$$  \hspace{1cm} (39)

where $H$ denotes the high-pass filter annihilating (approximately) the low-pass signal $f$. The low-pass signal is then estimated as

$$\hat{f} = L(y - \hat{x}_1 - \hat{x}_2) = y - \hat{x}_1 - \hat{x}_2 - H(y - \hat{x}_1 - \hat{x}_2),$$

where $L$ denotes the low-pass filter given by $L = I - H$.

As in Sec. II, we define the filter $H$ as $H = BA^{-1}$, where $A$ and $B$ are banded (see Sec. VI of [34]). We assume $B$ can be factored as $B = B_1D$. Because $D$ and $A^{-1}$ are matrix representations of LTI systems, they approximately commute. Hence, we formulate the problem as

---

**Fig. 3.** Reduction of transient artifacts in a NIRS time series using LPF/CSD with the atan penalty. (a) Raw data. (b, c, d) Output of the LPF/CSD problem with $(\lambda_0, \lambda_1)$ given by $(\lambda_0^*, 0)$, $(0, \lambda_1^*)$, and (36). (e) Corrected data (CSD signal (d) subtracted from raw data).

**Fig. 4.** Comparison of convergence behavior of ADMM and MM algorithms for the LPF/CSD problem.
\[ \{ \hat{x}_1, \hat{x}_2 \} = \arg \min_{x_1, x_2} \left\{ \frac{1}{2} \| H y - B A^{-1} x_1 - B_1 A^{-1} D x_2 \|_2^2 + \lambda_0 \sum_n \phi_0([x_1]_n) + \lambda_1 \sum_n \phi_1([D x_1]_n) + \lambda_2 \sum_n \phi_2([D x_2]_n) \right\}. \quad (40) \]

The advantage of this form compared to (39) is that it leads to an optimization algorithm that involves banded matrices exclusively, and hence admits a fast implementation.

Note that \( x_2 \) is not uniquely determined, because adding a constant (dc offset) to \( x_2 \) does not change the value of the objective function. Hence, it is sufficient to solve for the derivative (first-order difference), which we then integrate to obtain \( x_2 \). We denote the discrete-time integrator by \( S \) (defined such that \( DS = I \), as in [34]), and make the change of variables

\[ x_1 = A u_1, \quad D x_2 = A u_2. \quad (41) \]

Then the data fidelity term in (40) can be written as

\[ F_1(u_1, u_2) = \frac{1}{2} \| H y - B u_1 - B_1 u_2 \|_2^2 \quad (42) \]

and the regularization term in (40) can be written as

\[ F_2(u_1, u_2) = \lambda_0 \sum_n \phi_0([A u_1]_n) + \lambda_1 \sum_n \phi_1([D A u_1]_n) + \lambda_2 \sum_n \phi_2([A u_2]_n). \quad (43) \]

Then \( F = F_1 + F_2 \), and in terms of \( u_1 \) and \( u_2 \) we have the optimization problem

\[ \{ \hat{u}_0, \hat{u}_1 \} = \arg \min_{u_1, u_2} \left\{ F_1(u_1, u_2) + F_2(u_1, u_2) \right\}. \quad (44) \]

A. Algorithm

To solve (44), we use the majorization-minimization principle. We majorize \( F_1 \) with \( G_1 \), and \( F_2 \) by \( G_2 \). Then \( F \) is majorized with \( G = G_1 + G_2 \). Following the MM principle, we then iteratively minimize \( G \). With suitably chosen majorizers \( G_i \), the minimization of \( G \) can be implemented with high computational efficiency. After the minimizers \( \hat{u}_0 \) and \( \hat{u}_1 \) are obtained, we compute \( \hat{x}_1 \) and \( \hat{x}_2 \) as \( \hat{x}_1 = A \hat{u}_1 \) and \( \hat{x}_2 = S A \hat{u}_2 \), in accordance with (41).

To facilitate the following derivation, we define

\[ u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \quad v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}. \quad (45) \]

We then write the data fidelity term, \( F_1 \), as

\[ F_1(u) = \frac{1}{2} \| H y - [B \quad B_1] u \|_2^2. \quad (46) \]

The matrix of the quadratic term of \( F_1 \) (i.e., \( C \) in \( u^T C u \) after expansion of \( F_1(u) \)) is not banded. Hence, the use of direct minimization by fast banded solvers is precluded. However, a quadratic majorizer of \( F_1 \) is given by

\[ G_1(u, v) = F_1(u) + \frac{1}{2} (u - v)^T P (u - v), \quad (47) \]

where \( P \) is a positive semidefinite matrix. Note that \( G_1(u, u) = F_1(u) \). Since \( P \) is positive semidefinite,
We also have
\[ \nabla_u G_2(u, v) = \begin{bmatrix} R_1(v_1) & 0 \\ 0 & R_2(v_2) \end{bmatrix} u. \] (56)
Hence, \( \nabla_u G = \nabla_u G_1 + \nabla_u G_2 = 0 \) leads to the linear system
\[ \begin{bmatrix} 2B^T B + R_1(v_1) & 0 \\ 0 & 2B_1^T B_1 + R_2(v_2) \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} B^T Hy + B_1^T y \\ B_1^T y - B_1 u_2 \end{bmatrix}. \] (57)
Note that the system matrix is banded. The solution to (57) is given by
\[ g = Bu_1 - B_1 v_2 \] \[ u_1 = \left( 2B^T B + R_1(v_1) \right)^{-1} \left( B^T Hy + B_1^T g \right) \] (58) (59)
\[ u_2 = \left( 2B_1^T B_1 + R_2(v_2) \right)^{-1} \left( B_1^T Hy - B_1^T g \right). \] (60)

Hence, minimizing the majorizer \( G \) according to the MM update equation (17) leads to
\[ g^{(i)} = Bu_1^{(i)} - B_1 u_2^{(i)} \] (61)
\[ u_1^{(i+1)} = \left( 2B^T B + R_1(u_1^{(i)}) \right)^{-1} \left( B^T Hy + B_1^T g^{(i)} \right) \] (62)
\[ u_2^{(i+1)} = \left( 2B_1^T B_1 + R_2(u_2^{(i)}) \right)^{-1} \left( B_1^T Hy - B_1^T g^{(i)} \right). \] (63)

Equations (61)-(63) constitute the iterative algorithm, TARA, summarized in Table III. Note that the system matrices are banded; hence, the algorithm can be implemented using fast solvers for banded systems. The vectors \( B^T Hy \) and \( B_1^T Hy \) need to be computed one time only. The algorithm does not require any parameters other than the ones in (40). After \( u_1 \) and \( u_2 \) are obtained upon convergence of the algorithm, \( x_1 \) and \( x_2 \) are obtained using (41).

Run times of the new LPF/CSD algorithm and of TARA are shown in Fig. 5, as measured using a 2013 MacBook Pro (2.5 GHz Intel Core i5) running Matlab R2011a. The algorithms were run with a second order filter \( H \) and for 50 iterations. The run times are linear in the signal length, \( N \).

### B. Parameters

To use TARA (i.e., to solve (40)), three regularization parameters \( \lambda_i \) must be specified; if non-convex penalties are utilized, then three non-convexity parameters \( a_i \) must be specified as well. If \( \lambda_2 \) is appropriately set (i.e., such that \( x_2 \approx x_2 \)), then \( y - \hat{x}_2 \) can be modeled as having Type 1 artifacts only, and \( \lambda_0 \) and \( \lambda_1 \) can be nominally set as in Sec. II-C. In the examples below, we set \( \lambda_0 \) and \( \lambda_1 \) using (36) with \( \theta \in (0, 1) \) manually tuned so that \( \hat{x}_0 \) adheres to a baseline value of zero.

The \( \lambda_2 \) parameter should be set, in part, according to the noise variance. A constraint on \( \lambda_2 \) is obtained via the noise analysis described in II-C. Namely, we obtain the same nominal value (35) as for \( \lambda_2^* \). Hence, to prevent noise contamination of \( x_2 \), we require \( \lambda_2 > \lambda_2^* \).

The Type 1 and Type 2 artifacts are implicitly and operationally defined through the \( \lambda_i \) parameters. Their relative values affect the apportionment of signal features between \( \hat{x}_1 \) and \( \hat{x}_2 \). However, the total artifact signal \( \hat{x}_1 + \hat{x}_2 \), and \( \hat{f} \), are quite robust to small changes in \( \lambda_i \), as will be illustrated in the Example in Sec. III-C.

We note that it is reasonable to set \( \lambda_2 > \lambda_1 \). Because, if we set \( \lambda_2 \leq \lambda_1 \) and \( \lambda_0 > 0 \) in (40), then \( x_1 \) is strictly more regularized than \( x_2 \) and, consequently, the solution \( x_1 \) in (40) will always be identically zero. It can be said that setting \( \lambda_i \) in this way leads to all the transient artifacts being classified as Type 2. In this case, the \( x_1 \) component may as well be omitted from the problem formulation.

The parameter \( \lambda_2 \) may be set so as to distinguish between Type 1 and Type 2 artifacts. To consider how \( \lambda_2 \) influences the implicit distinction between Type 1 and Type 2 artifacts, consider problem (40) with \( \ell_1 \) norm penalties,
\[ F(x_1, x_2) = \frac{1}{2} \| H(y - x_1 - x_2) \|_2^2 + \lambda_0 \| x_1 \|_1 + \lambda_1 \| D x_1 \|_1 + \lambda_2 \| D x_2 \|_1. \] (64)
Suppose \( y \) is all zero except for some transient pulse that we consider a Type 1 artifact. It can be expected that the
minimizer of \( F \) is likewise some transient pulse, which we denote by \( p \); i.e., \( \hat{x}_1 + \hat{x}_2 = p \). If \( p \) is considered a Type 1 artifact, then \( \lambda_1 \) should be set so that \( \hat{x}_1 = p \) and \( \hat{x}_2 = 0 \). To find a rule for setting the parameter values, we evaluate the objective function \( F \) for two candidate solutions:

\[
S_1 = \{ x_1 = p, \ x_2 = 0 \}, \quad S_2 = \{ x_1 = 0, \ x_2 = p \}. \tag{65}
\]

If \( S_1 \) minimizes \( F \), then TARA correctly classifies \( p \) as a Type 1 artifact; while if \( S_2 \) minimizes \( F \), then TARA incorrectly classifies \( p \) as a Type 2 artifact. Solution \( S_1 \) can be the optimal solution only if \( F(S_1) < F(S_2) \). Because \( (x_1 + x_2) \) is the same for solutions \( S_1 \) and \( S_2 \), the data fidelity term of \( F \) is equal for \( S_1 \) and \( S_2 \). Hence, the relative cost depends only on the penalty terms. Therefore, we have

\[
\lambda_0 \| p \|_1 + \lambda_1 \| Dp \|_1 \leq \frac{S_1}{S_2} \lambda_2 \| Dp \|_1 \tag{66}
\]

or

\[
\lambda_0 \frac{\| p \|_1}{\| Dp \|_1} + \lambda_1 \frac{S_1}{S_2} \leq \lambda_2. \tag{67}
\]

The notation \( \leq \) means \( S_1 \) is the optimal solution if the left-hand side is the smaller value, while \( S_2 \) is optimal if the right-hand side is smaller. Hence, for \( p \) to be classified as a Type 1 artifact by TARA, \( \lambda_2 \) must be at least as great as the quantity on the left-hand side of (67). Note that condition (67) is invariant to amplitude scaling of \( p \); i.e., only the shape of \( p \) matters.

As an example, suppose that \( p \) is taken to be a rectangular pulse of length \( M \) samples and amplitude \( A \). Then \( \| p \|_1 = MA \) and \( \| Dp \|_1 = 2A \), so condition (67) can be written as

\[
0.5M\lambda_0 + \lambda_1 \frac{S_1}{S_2} \leq \lambda_2. \tag{68}
\]

Hence, for the \( M \)-point pulse to be exhibited in \( x_1 \), the parameter \( \lambda_2 \) must exceed \( \lambda_1 \) by \( 0.5M\lambda_0 \). When \( (\lambda_0, \lambda_1) = (\theta \lambda_0^*, (1 - \theta)\lambda_0^*) \), as suggested in (36), then (68) gives a condition in terms of \( \theta \),

\[
\theta(0.5M\lambda_0^* - \lambda_1^*) + \lambda_1^* \frac{S_1}{S_2} \leq \lambda_2. \tag{69}
\]

We noted above that \( \lambda_2 \) should satisfy \( \lambda_2 \geq \lambda_1^* \). Hence, \( \lambda_2 \) should be set according to

\[
\lambda_2 \geq \max \{ \lambda_1^*, \lambda_1^* + \theta(0.5M\lambda_0^* - \lambda_1^*) \}. \tag{70}
\]

It experiments, we have found that \( 0.5M\lambda_0^* - \lambda_1^* \) is usually positive, so the second term in (70) dominates. Moreover, \( \theta \) is often relatively small in practice (sufficient so that \( x_1 \) adheres to a baseline of zero). Hence, it will often be sufficient that \( \lambda_2 \) be only slightly larger than \( \lambda_1^* \). The use of condition (68) to control the behavior of TARA is illustrated in Sec. III-C.

Based on the foregoing considerations, we suggest writing \( \lambda_2 = \beta \lambda_1^* \) and taking \( \beta \) as a tuning parameter with a nominal range of \( \beta \in [1, 2] \). In conjunction with the discussion in Sec. II-D, we obtain a parameterization of the TARA problem (40) in terms of \( (\sigma, \theta, \beta) \) instead of \( (\lambda_0, \lambda_1, \lambda_2) \). We find this parameterization more useful in practice because the influence of each parameter can be more readily understood.

\[\text{Fig. 6. Signal decomposition and filtering with TARA. (a) Pulses 4 samples and shorter appear in } x_1. \text{ (b) Pulses 4 samples and longer appear in } x_2.\]

In particular, we consider \( (\theta, \beta) \) to be shape parameters; they influence the shape of the estimated transients.

When non-convex penalties are utilized, \( a_0 \) and \( a_1 \) can be set as in (37). Following the same considerations as in Sec. II-E, we set \( a_2 = 0.5 \| h_1 \|_2^2 / \lambda_2 \) like for \( a_1 \).

C. TARA Example 1

This example shows TARA as applied to the simulated time series \( y \) shown in Fig. 6. The signal consists of two
low-frequency sinusoids, several additive rectangular pulses of short duration, several additive step discontinuities, and additive white Gaussian noise (σ = 0.3). Each of the rectangular pulses has a length of four samples, except for the last pulse (at n = 150) which has a length of three samples.

In this example, we use a fourth-order zero-phase Butterworth filter with $f_c = 0.03$ cycles/sample. We also use the non-convex atan penalty, set $\lambda_0^*$ and $\lambda_1^*$ according to (31) and (35) in Sec. II-C, and set $\theta = 0.3$ by manual tuning as in Sec. II-F.

To demonstrate the influence of $\lambda_2$ as discussed in Sec. III-B, we consider the question of how to set $\lambda_2$ to ensure that the brief rectangular pulses appear in $\hat{x}_1$ rather than in $\hat{x}_2$ (i.e., to ensure TARA classifies these pulses as Type 1 artifacts). Since all the brief pulses are of length 4 or less, we set $M = 4$ in (68) to find that $2\lambda_0 + \lambda_1$ is the critical value for $\lambda_2$. Hence, we must set $\lambda_2 > 2\lambda_0 + \lambda_1$ to ensure that the pulses appear in $\hat{x}_1$. Therefore, we set $\lambda_2 = 1.1 \times (2\lambda_0 + \lambda_1)$; i.e., $\beta = 1.1$. The output of TARA for this $\lambda_2$ is shown in Fig. 6(a). In conformity with our expectation, all the brief rectangular pulses appear in $\hat{x}_1$, i.e., they are classified by TARA as Type 1 artifacts. The signal $\hat{x}_2$ is piecewise constant and contains no brief pulses. (The small step at the end of $\hat{x}_1$ is a boundary artifact due to applying the recursive filter $\mathbf{H}$ to a finite-length signal.)

To further illustrate the role of $\lambda_2$, we set $\lambda_2 = 0.9 \times (2\lambda_0 + \lambda_1)$. This value is less than the critical value needed to classify a length-4 pulse as a Type 1 artifact. Accordingly, it is expected that TARA will classify pulses of length 4 and longer as Type 2 artifacts and that they will be exhibited in $\hat{x}_2$. The output of TARA for this value of $\lambda_2$ is shown in Fig. 6(b). As predicted, the length-4 pulses are exhibited in $\hat{x}_2$. The only pulse exhibited in $\hat{x}_1$ is the final one (at $n = 150$), which is of length 3. This example validates the use of $\lambda_2$ for the disambiguation of pulses based on their duration.

This example uses rectangular pulses because of the availability of the simple formula (68). TARA does not explicitly model a signal in terms of rectangular pulses, and its effectiveness is not limited to rectangular artifacts. For real data with transient artifacts of complex shape, it is not expected that a simple formula for a critical value will be available; however, the general influence of $\lambda_2$ on the relative properties of $\hat{x}_1$ and $\hat{x}_2$ holds. Namely, decreasing $\lambda_2$ results in more waveforms being classified as Type 2 artifacts.

Note that the low-pass signal, $\hat{f}$, is essentially the same as in Figs. 6(a) and 6(b). Likewise, the total signal, $\hat{x} = \hat{x}_1 + \hat{x}_2 + \hat{f}$, is approximately the same in both figures. The small change in $\lambda_2$ produced only a small change in the total signal, even though it produced a large change in $\hat{x}_1$. Hence, for the purpose of denoising, the total signal is not overly sensitive to the exact value of $\lambda_2$. Note that TARA provides a reasonable denoising result: the total signal is relatively noise-free, preserves the discontinuities in the data, and does not exhibit ringing around the discontinuities.

Since wavelet methods have been successfully used for the correction of transient artifacts [9], [21], [24], we illustrate a wavelet-based decomposition in Fig. 7 of the same example signal as was considered in Fig. 6. We use the stationary (un-decimated) wavelet transform [11] with the Haar wavelet filter and the non-negative garrote threshold function [19] (same as in [21]). As shown, the reconstruction of the signal from the thresholded wavelet coefficients (excluding the low-pass component) adheres to a baseline of zero, captures the transient pulses, and is approximately noise-free. The reconstruction of the signal from the low-pass (non-thresholded) wavelet coefficients is a smooth noise-free signal. However, using wavelet transforms, the step changes (Type 2 transients) cannot be easily isolated from the rest of the signal, because the step discontinuities are represented by both high-frequency and low-frequency wavelet coefficients. For example, the step discontinuity at time index 40 appears as a bi-phasic pulse in the wavelet-reconstructed signal, and the change in baseline value induced by the step discontinuity is absorbed into the low-pass component. As a consequence, the wavelet estimate of the signal exhibits spurious ripples and cusps where the signal has discontinuities. In addition, the low-pass component of the wavelet decomposition less accurately recovers the low-pass component of the data. In this example, the low-pass signal as estimated by the wavelet method and TARA have root-mean-squared errors of 0.85 and 0.29, respectively; i.e., TARA recovers the low-pass component more accurately.

D. TARA Example 2

This example illustrates TARA as applied to the NIRS time series shown in Fig. 8, which comes from the same measurement that produced the data used shown in Fig. 3. The time series was acquired using a pair of optodes on the back of a subject’s head; as such, the data is susceptible to motion artifacts in the form of abrupt shifts of the baseline. A prominent baseline shift can be seen at time index 470 in Fig. 8(a). Other motion artifacts also are visible. This data was also used in [34].
To apply TARA for artifact suppression, we must specify the filter $H$, the three regularization parameters, and the penalty functions. We used a second-order zero-phase Butterworth filter with $f_c = 0.06$ cycles/sample. The tuning procedure described in Sec. II-D was used to set $\lambda_0^0$ and $\lambda_1^0$ with a pseudo-noise standard deviation of $\sigma = 0.25$. We manually tuned the shape parameters to $\theta = 0.05$ and $\beta = 1.4$. The atan penalty was used, with non-convexity parameters set according to (37). We ran TARA for 100 iterations with a run time of about 0.28 seconds.

The Type 1 and Type 2 artifact signals estimated by TARA, $\hat{x}_1$ and $\hat{x}_2$, shown in Fig. 8, are sparse and approximately piecewise constant, as intended. The estimated total artifact signal, $\hat{x}_1 + \hat{x}_2$, which comprises additive step discontinuities and transient spikes, appears to accurately model the artifacts present in the data. Note that the corrected time series, obtained by subtracting the total estimated artifact signal from the original time series, has both low-frequency and high-frequency spectral content. Compared with [34], in which LPF/TVD processing is applied to the same data, the artifacts appear to be more accurately estimated here.

E. Wavelet-based artifact reduction

It has been found that wavelet methods compare favorably to other methods for the correction of motion artifacts in single-channel NIRS time series [9], [21], [24]. Figure 9 compares wavelet transient artifact reduction (WATAR) and TARA as applied to the NIRS data from Fig. 8. As in [21], we use the stationary (un-decimated) wavelet transform [11] with the Haar wavelet filter and the non-negative garrote threshold function [19]. We apply thresholding to all subbands except the low-pass one. The wavelet-corrected time series does not have the long-term drift that the TARA-corrected time series has; however, that is easily removed by LTI filtering and its removal is not an objective of TARA. Moreover, some biological information may be present at low frequencies (see Sec. III-G).

It can be seen that both methods otherwise give generally similar results, but the TARA-estimated artifact signal captures abrupt changes in the data, unlike the wavelet-estimated artifact signal. The artifact in the interval 1370-1420 is estimated by TARA with distinct pre- and post-artifact baseline values, while the wavelet-estimated artifact signal exhibits a small change due to the slowly-varying low-pass component (coming from the low-pass subband of the wavelet transform). In addition, TARA finds an abrupt change at time index 1530, while the wavelet method exhibits only a small bi-phasic (zero-mean) pulse at that instant. TARA is better able to estimate abrupt step changes than the wavelet method because it is explicitly based on a two-component model. In NIRS time series analysis, motion artifacts often cause step changes, and this motivates the accurate estimation thereof.

That TARA and wavelet methods give similar results can be explained by their being based on similar underlying models. The wavelet method implicitly models transient artifacts as piecewise smooth. TARA is based on a similar model, but uses an optimization approach instead of a fixed transform.

F. Multichannel data

Physiological time-series data (e.g., NIRS, EEG) are often acquired in multichannel form. If different regularization parameters are to be required for each channel, then setting the parameters will be a problematic issue. In this example, we apply TARA to multichannel data (Fig. 10, black) and use the same shape parameters ($\alpha$, $\beta$) and filter $H$ for all channels. The pseudo-noise parameter $\sigma$ should, however, be set channel-by-channel, because the channels are not equally normalized.
For this illustration, we set $\sigma$ for each channel based on the artifact-free segment of the data extending from 550 to 950 time frames (see Fig. 10). The $\sigma$ values were set by applying a high-pass filter to each channel, and then computing the standard deviation of its output in the artifact-free segment.

The artifacts and corrected data obtained using TARA are shown in Fig. 10. For the purpose of display, to avoid cumulative baseline drift, each corrected time series has been filtered with a zero-phase second-order recursive dc-notch filter. TARA effectively reduces transient artifacts in most channels, without introducing substantial distortion. Some channels of the corrected data exhibit slow waves around the removed transients; these are consistent with the assumed

signal model, wherein the signal is modeled as comprising a low-pass signal. This example suggests that with a fixed pair of shape parameter values ($\alpha$, $\beta$) and filter $H$, TARA may be quite effective for multiple channels and that, given an artifact-free segment (obtained here by visual inspection), it may be possible to use TARA in a data-driven automated manner.

G. Preservation of hemodynamic response

In the course of suppressing artifacts, biological information of interest should not be distorted or attenuated. For NIRS specifically, any hemodynamic response (HR) waveforms present should be preserved. Equivalently, the artifact signals should not be affected by HRs, because the HRs should not be mistaken for artifacts. To test TARA in this regard, we add a simulated HR to each channel of the considered multichannel data. The data with the simulated HR is shown in gray in Fig. 10. The artifact signals and corrected data are likewise shown in gray. The gray-colored artifact signals, obtained from the HR-added data, are nearly indistinguishable from the original artifact signals. Accordingly, TARA accurately preserves the HR in the corrected data.

To compare TARA and WATAR (i.e., wavelets), we likewise apply WATAR to the same multichannel data, with and without the added HR. The WATAR-estimated artifact signals (Fig. 11) exhibit a noticeable portion of the HR in about half the channels. Consequently, the HR is more attenuated and distorted.
in the WATAR-corrected data than in the TARA-corrected data. In quantitative terms, we measure the root-mean-squared deviation over the HR interval (700-880) between the HR and non-HR artifact signals. The value is 0.18 for TARA and 0.53 for WATAR. By this measure, WATAR is affected by the HR 2.9 times more than TARA, and so TARA better preserves the HR than WATAR. This is intended as an illustrative example; a more thorough investigation such as in [15], [21], [31] is needed before definitive statements can be made.

IV. Conclusion

For the purpose of identifying and isolating transient artifacts in time series, this work distinguishes two types of artifact signals: one type that consists of infrequent transient pulses and otherwise adheres to a baseline value of zero, and a second type which consists of abrupt shifts of the baseline (i.e., additive step discontinuities). In this work, the observed time series is modeled as the sum of an artifact signal of each type, a low-pass signal (e.g., a background trend), and a white Gaussian stochastic process. To jointly estimate the components of the signal model, we formulate an optimization problem and develop a rapidly converging, computationally efficient iterative algorithm, denoted TARA (for ‘transient artifact reduction algorithm’). We also address the selection of the regularization and non-convexity parameters. The effectiveness of the approach is illustrated using both simulated and NIRS time-series data. The presented approach is also useful for denoising piecewise smooth signals (Figs. 1 and 6).

Some artifacts arising in biomedical time series, such as oscillatory transients, are not of the form considered here. In this case, the approach of TARA should be modified to account for the artifact characteristics, or other methods should be used, e.g., [15], [21]. Additionally, if the available data is corrupted by high-amplitude oscillatory noise or interference, then the Gaussian noise assumption is not satisfied, and the approach should be modified accordingly.

To exploit TARA for two-dimensional (2-D) data, one issue is that the sparse filter matrices (A and B) may not be exactly banded, depending on how the 2-D case is formulated. In that case, fast solvers for banded systems may not be so readily utilized. Hence, the high computational efficiency of TARA might not completely carry over to the 2-D case. As a result, an extension of TARA to the 2-D case remains of interest as future work.

ACKNOWLEDGEMENT

The authors gratefully acknowledge Justin R. Estepp (Air Force Research Laboratory, Wright-Patterson AFB, OH) and Sean M. Weston (Oak Ridge Institute for Science and Education, TN) for experimental NIRS data. The authors also thank the reviewers for their constructive comments.

REFERENCES


function \[x, f, \text{cost}\] = lpfcsd(y, d, fc, lam0, lam1, pen, a0, a1, Nit, x_init)
% \[x, f, \text{cost}\] = lpfcsd(y, d, fc, lam0, lam1, pen, a0, a1, Nit)
% Simultaneous low-pass filtering and compound sparsity denoising.

% INPUT
% y - ras data
% d - degree of filter is 2d (d = 1, 2, 3)
% fc - cut-off frequency (normalized frequency, 0 < fc < 0.5)
% lam0, lam1 - regularization parameters for x and diff(x)
% pen - penalty function (’L1’, ’log’, or ’atan’)
% a0, a1 - non-convexity parameters (ignored for ’L1’ penalty)
% Nit - number of iterations

% OUTPUT
% x - CSD component
% f - LPF component
% cost - cost function history
% Use lpfcsd(..., x_init) to specify initial x
% Objective function uses smoothed penalty. Algorithm is MM.

% Ivan Selesnick
% NYU Polytechnic School of Engineering, New York, USA
% September 2013
% revised February 2014

% Use smoothed penalty functions
EPS = 1E-8;
switch pen
  case ’L1’
    phi = @(x, a) sqrt(x.^2 + EPS);
    psi = @(x, a) sqrt(x.^2 + EPS);
    a0 = 0;
    a1 = 0;
  case ’log’
    phi = @(x, a) (1/a) * log(1 + a*sqrt(x.^2 + EPS));
    psi = @(x, a) sqrt(x.^2 + EPS) .* (1 + a*sqrt(x.^2 + EPS));
  case ’atan’
    phi = @(x, a) 2./(a*sqrt(3)) .* (atan((2*a.*sqrt(x.^2 + EPS)+1)/sqrt(3)) - pi/6);
    psi = @(x, a) sqrt(x.^2 + EPS) .* (1 + a.*sqrt(x.^2 + EPS) + a.^2.*(x.^2 + EPS));
  otherwise
    disp(’Error: penalty must be L1, log, or atan’) x = []; f = []; cost = [];
    return
end

cost = zeros(1, Nit); % cost function history
y = y(:); % convert to column vector
N = length(y);
[A, B] = BAfilt(d, fc, N);
Hy = B*(A\y);
BTB = B'*B;
b = BTB*(A\y);
e = ones(N, 1);
D = spdiags([-e, e], [0 1], N-1, N);

if exist(’x_init’, ’var’) % initialization
  x = x_init;
else
  x = y;
end

for k = 1:Nit
  Lam0 = spdiags(lam0./psi(x, a0), 0, N, N);
Lam1 = spdiags(lam1./psi(D*x, a1), 0, N-1, N-1);
Q = BTB + A' * (Lam0 + D'*Lam1*D) * A;
u = Q \ b;
x = A \ u; % Note: H \ x = B \ (A\x) = B \ u

\text{cost}(k) = lam0 \ \text{sum}(\phi(x, a0)) + lam1 \ \text{sum}(\phi(D*x, a1)) + 0.5 \ \text{sum}(abs(B*u-Hy)^2);
end

bn = nan(d, 1); % bn : nan’s to extend f to length N
f = y - x - [bn; Hy - B*u; bn]; % f : low-pass component

function [x1, x2, f, cost] = tara_L1(y, d, fc, lam0, lam1, lam2, Nit)
% [x1, x2, f, cost] = tara_L1(y, d, fc, lam0, lam1, lam2, Nit)
% Transient Artifact Reduction Algorithm (TARA)
% with L1 norm penalty
% INPUT
% y - raw data
% d - degree of filter is 2d (d = 1, 2, 3)
% fc - cut-off frequency (normalized frequency, 0 < fc < 0.5)
% lam0, lam1 - regularization parameter for x1 and diff(x1)
% lam2 - regularization parameter for diff(x2)
% Nit - number of iterations
% OUTPUT
% x1 - sparse signal with sparse derivative
% x2 - signal with sparse derivative
% cost - cost function history

y = y(:);
N = length(y);
cost = zeros(1, Nit);
EPS = 1E-10;
phi = @(x) sqrt(x.^2 + EPS);

[A, B, B1] = BAfilt(d, fc, N);
A1 = A(1:N-1, 1:N-1);
e = ones(N, 1);
D = spdiags([-e, e], 0:1, N-1, N);
BTB = B'*B;
B1TB1 = B1'*B1;
Hy = B * ( A \ y );
y1 = B' * Hy;
y2 = B1' * Hy;

u1 = zeros(N, 1);
u2 = zeros(N-1, 1);
x1 = zeros(N, 1);
Dx2 = zeros(N-1, 1);

for i = 1:Nit
Lam0 = spdiags(lam0./phi(x1), 0, N, N);
Lam1 = spdiags(lam1./phi(D*x1), 0, N-1, N-1);
Lam2 = spdiags(lam2./phi(Dx2), 0, N-1, N-1);

Q1 = 2*BTB + A' * (Lam0 + D' * Lam1*D) * A;
Q2 = 2*B1TB1 + A1' * Lam2 * A1;
g = B * u1 - B1 * u2;
u1 = Q1 \ (y1 + B' * g);
u2 = Q2 \ (y2 - B1' * g);

x1 = A * u1;
Dx2 = A1 * u2;
cost(i) = 0.5 * sum((Hy - B*u1 - B1*u2).^2) + ...
\begin{align*}
& \text{lam0} \cdot \text{sum}(\phi(x_1)) + \text{lam1} \cdot \text{sum}(\phi(D \cdot x_1)) + \text{lam2} \cdot \text{sum}(\phi(Dx_2)); \\
& \text{end}
\end{align*}

\begin{align*}
x_2 &= [0; \text{cumsum}(Dx_2)]; \\
bn &= \text{nan}(d, 1); \\
f &= y - x_1 - x_2 - [bn; Hy - B*u1 - B1*u2; bn]; \quad \% f : \text{low-pass component}
\end{align*}

\begin{verbatim}
function [x1, x2, f, cost, u1, u2] = tara(y, d, fc, lam0, lam1, lam2, pen, a0, a1, a2, Nit, u1_init, u2_init)
% tara(...) = tara(y, d, fc, lam0, lam1, lam2, pen, a0, a1, a2, Nit)
% Transient Artifact Reduction Algorithm (TARA)
%
% INPUT
% y - raw data
% d - degree of filter is 2d (d = 1, 2, 3)
% fc - cut-off frequency (normalized frequency, 0 < fc < 0.5)
% lam0, lam1 - regularization parameter for x1 and diff(x1)
% lam2 - regularization parameter for diff(x2)
% pen - penalty function ('L1', 'log', or 'atan')
% a0, a1, a2 - non-convexity parameters (ignored for 'L1' penalty)
% Nit - number of iterations
%
% OUTPUT
% x1 - sparse signal with sparse derivative
% x2 - signal with sparse derivative
% cost - cost function history
%
% Use tara(..., u1_init, u2_init) to specify initial vectors

y = y(:); \\
N = length(y); \\
cost = zeros(1, Nit);

% Use smoothed penalty functions
EPS = 1E-10; \\
% EPS = 1E-8;
switch pen
  case 'L1'
    phi = @(x, a) sqrt(x.^2 + EPS); \\
    psi = @(x, a) sqrt(x.^2 + EPS); \\
    a0 = 0; a1 = 0; a2 = 0;
  case 'log'
    phi = @(x, a) (1/a) * log(1 + a*sqrt(x.^2 + EPS)); \\
    psi = @(x, a) sqrt(x.^2 + EPS) .* (1 + a.*sqrt(x.^2 + EPS));
  case 'atan'
    phi = @(x, a) 2./(a*sqrt(3)) .* (atan(2*a.*sqrt(x.^2 + EPS)+1)/sqrt(3)) - pi/6; \\
    psi = @(x, a) sqrt(x.^2 + EPS) .* (1 + a.*sqrt(x.^2 + EPS) + a.^2.*(x.^2 + EPS));
  otherwise
    disp('Error: penalty must be L1, log, or atan')
    x1 = []; x2 = []; f = []; cost = [];
    return
\end{verbatim}

\begin{verbatim}
[A, B, B1] = BAfilt(d, fc, N); \\
A1 = A(1:N-1, 1:N-1); \\
e = ones(N, 1); \\
D = spdiags([-e, e], 0:1, N-1, N); \\
BTB = B'*B; \\
BITB1 = B1'*B1; \\

Hy = B + ( A \ y ); \\
y1 = B' * Hy; \\
y2 = B1' * Hy;

% initialization
if exist('u1_init', 'var')
  u1 = u1_init;
else
\end{verbatim}
if exist('u2_init', 'var')
    u2 = u2_init;
else
    u2 = zeros(N-1, 1);
end

x1 = A * u1;
Dx2 = A1 * u2;

for i = 1:Nit
    Lam0 = spdiags( lam0./psi(x1, a0), 0, N, N);
    Lam1 = spdiags( lam1./psi(D*x1, a1), 0, N-1, N-1);
    Lam2 = spdiags( lam2./psi(Dx2, a2), 0, N-1, N-1);
    Q1 = 2*BTB + A' * ( Lam0 + D' * Lam1 * D ) * A;
    Q2 = 2*B1TB1 + A1' * Lam2 * A1;
    g = B * u1 - B1 * u2;
    u1 = Q1 \
        ( y1 + B' * g);
    u2 = Q2 \
        ( y2 - B1' + g);
    x1 = A * u1;
    Dx2 = A1 * u2;
    cost(i) = 0.5 * sum((Hy - B*u1 - B1*u2).^2) + ...
        lam0 * sum(phi(x1, a0)) + lam1 * sum(phi(D*x1, a1)) + lam2 * sum(phi(Dx2, a2));
end

x2 = [0; cumsum(Dx2)];
bn = nan(d, 1);

f = y - x1 - x2 - [bn; Hy - B*u1 - B1*u2; bn];  % f : low-pass component