Translation-invariant shrinkage/thresholding of group sparse signals.
P.-Y. Chen and I. W. Selesnick.
Signal Processing
Vol. 94, pp 476-489, January 2014.

Translation-Invariant Shrinkage/Thresholding of Group Sparse Signals

Po-Yu Chen and Ivan W. Selesnick
Polytechnic Institute of New York University, 6 Metrotech Center, Brooklyn, NY 11201, USA
Email: poyupaulchen@gmail.com, selesi@poly.edu. Tel: +1 718 260-3416.

March 20, 2013

Abstract

This paper addresses signal denoising when large-amplitude coefficients form clusters (groups). The L1-norm and other separable sparsity models do not capture the tendency of coefficients to cluster (group sparsity). This work develops an algorithm, called 'overlapping group shrinkage' (OGS), based on the minimization of a convex cost function involving a group-sparsity promoting penalty function. The groups are fully overlapping so the denoising method is translation-invariant and blocking artifacts are avoided. Based on the principle of majorization-minimization (MM), we derive a simple iterative minimization algorithm that reduces the cost function monotonically. A procedure for setting the regularization parameter, based on attenuating the noise to a specified level, is also described. The proposed approach is illustrated on speech enhancement, wherein the OGS approach is applied in the short-time Fourier transform (STFT) domain. The OGS algorithm produces denoised speech that is relatively free of musical noise.

1 Introduction

In recent years, many algorithms based on sparsity have been developed for signal denoising, deconvolution, restoration, and reconstruction, etc. [23]. These algorithms often utilize nonlinear scalar shrinkage/thresholding functions of various forms which have been devised so as to obtain sparse representations. Examples of such functions are the hard and soft thresholding functions [22], and the nonnegative garrote [29, 30]. Numerous other scalar shrinkage/thresholding functions have been derived as MAP or MMSE estimators using various probability models, e.g. [25, 37, 46].

We address the problem of denoising, i.e., estimating $x(i)$, $i \in \mathcal{I}$, from noisy observations $y(i)$,

$$y(i) = x(i) + w(i), \qquad i \in \mathcal{I}, \tag{1}$$

where the signal $x(i)$ is known to have a group sparse property and $w(i)$ is white Gaussian noise. Here, $\mathcal{I}$ denotes the domain of x, typically $\mathcal{I} = \{0, \ldots, N-1\}$ for one-dimensional finite-length signals. A generally effective approach for deriving shrinkage/thresholding functions is to formulate the optimization problem

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \left\{ F(\mathbf{x}) = \frac{1}{2} \|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda R(\mathbf{x}) \right\} \tag{2}$$

This work is supported by the NSF under Grant No. CCF-1018020.

where $\mathbf{x} = (x_i, i \in \mathcal{I})$ is the signal to be determined from the observation $\mathbf{y} = (y_i, i \in \mathcal{I})$. In problem (2), $\mathbf{x}$ may represent either the coefficients of a signal (e.g. wavelet or short-time Fourier transform coefficients) or the signal itself, if the signal is well modeled as sparse. The penalty function, or regularizer, $R(\mathbf{x})$, should be chosen to promote the known behavior of $\mathbf{x}$. Many of the shrinkage/thresholding functions devised in the literature can be derived as solutions to (2), where $R(\mathbf{x})$ is specifically of the separable form

$$R(\mathbf{x}) = \sum_{i \in \mathcal{I}} r(x(i)), \qquad r : \mathbb{C} \to \mathbb{R}. \tag{3}$$

For example, soft-thresholding is derived as the solution to (2) where $r(x) = |x|$, which corresponds to the lasso problem [71] or the basis pursuit denoising problem [11]. When $R(\mathbf{x})$ has the form (3), the variables $x(i)$ in (2) are decoupled, and the optimization problem (2) is equivalent to a set of scalar optimization problems. Therefore, the separable form (3) significantly simplifies the task of solving (2), because in this case the optimal solution is obtained by applying a scalar shrinkage/thresholding function independently to each element $x(i)$ of $\mathbf{x}$. From a Bayesian viewpoint, the form (3) models the elements $x(i)$ as being statistically independent.

For many natural (physically arising) signals, the variables (signal/coefficients) $\mathbf{x}$ are not only sparse but also exhibit a clustering or grouping property. For example, wavelet coefficients generally have inter- and intra-scale clustering tendencies [43, 66, 70]. Likewise, the clustering/grouping property is also apparent in a typical speech spectrogram. In both cases, significant (large-amplitude) values of $\mathbf{x}$ tend not to be isolated.

This paper develops a simple translation-invariant shrinkage/thresholding algorithm that exploits the grouping/clustering properties of the signal/coefficient vector $\mathbf{x}$. The algorithm acts on $\mathbf{x}$ as a whole without performing block-by-block processing, and minimizes the cost function (2) with the (non-separable) penalty function

$$R(\mathbf{x}) = \sum_{i \in \mathcal{I}} \left[\sum_{j \in \mathcal{J}} |x(i+j)|^2 \right]^{1/2}, \tag{4}$$

where the set $\mathcal{J}$ defines the group. The algorithm is derived using the majorization-minimization (MM) method [28, 53]. The algorithm can be considered an extension of the successive substitution algorithm for multivariate thresholding [64] or as an extension of the FOCUSS algorithm [59].

The penalty function (4) has been considered previously, but existing algorithms for its minimization [2, 3, 20, 27, 39, 40, 54] are based on variable duplication (variable splitting, latent/auxilliary variables, etc.). The number of additional parameters is proportional to the overlap; hence their implementations require additional memory and accompanying data indexing. The iterative algorithm proposed here avoids variable duplication and can be efficiently implemented via separable convolutions.

For the purpose of denoising, this paper also develops a conceptually simple method to set the regularization parameter λ analogous to the 'three-sigma' rule. The method allows for λ to be selected so as to ensure that the noise variance is reduced to a specified fraction of its original value. This method does not aim to minimize the mean square error or any other measure involving the signal to be estimated, and is thus non-Bayesian. Although conceptually simple, the method for setting λ is analytically intractable due to the lack of explicit functional form of the estimator. However, with appropriate pre-computation, the method can be implemented by table look-up.

In Section 2, the cost function is given and the iterative successive substitution algorithm for its minimization is presented. In Section 3, the effect of the shrinkage/thresholding algorithm on white Gaussian noise is investigated and is used to devise a simple method for selecting the regularization parameter λ. Section 4

illustrates the proposed approach to signal denoising, including speech enhancement.

1.1 Related work

The penalty function (4) can be considered a type of mixed norm. Mixed norms and their variants can be used to describe non-overlapping group sparsity [24,41,42,75] or overlapping group sparsity [2,3,12,20,27,38, 39,51,54,74]. Algorithms derived using variable splitting and the alternating direction method of multipliers (ADMM) are described in [7, 20, 27]. These algorithms duplicate each variable for each group of which the variable is a member. The algorithms have guaranteed convergence, although additional memory is required due to the variable duplication (variable splitting). A theoretical study regarding recovery of group support is given in [38] which uses an algorithm that is also based on variable duplication. A more computationally efficient version of [38] for large data sets is described in [51] which is based on identifying active groups (non-zero variables). The identification of non-zero groups is also performed in [74] using an iterative process to reduce the problem size; a dual formulation is then derived which also involves auxiliary variables. Auxiliary and latent variables are also used in the algorithms described in [2, 3, 40, 54], which, like variable splitting, calls for additional memory proportional to the extent of the overlap. Overlapping groups are used to induce sparsity patterns in the wavelet domain in [61] also employing variable duplication.

Several other algorithmic approaches and models have been devised to take into account clustering or grouping behavior, such as: Markov models [10, 18, 21, 26, 31, 50, 55, 72], Gaussian scale mixtures [33, 56], neighborhood-based methods with locally adaptive shrinkage [8,49,67], and multivariate shrinkage/thresholding functions [1, 13, 57, 58, 62, 64, 65]. These algorithms depart somewhat from the variational approach wherein a cost function of the form (2) is minimized. For example, in many neighborhood-based and multivariate thresholding methods, local statistics are computed for a neighborhood/block around each coefficient, and then these statistics are used to estimate the coefficients in the neighborhood (or the center coefficient). In some methods, the coefficients are segmented into non-overlapping blocks and each block is estimated as a whole; however, in this case the processing is not translation-invariant and some coefficient clusters may straddle multiple blocks.

It should be noted that an alternative penalty function for promoting group sparsity is proposed in [38,52] (see equation (3) of [52]). Interestingly, it is designed to promote sparse solutions for which the significant values tend to be comprised of the unions of groups; while, as discussed in [38, 52], the penalty function (4) promotes sparse solutions for which the significant values tend to be comprised of the *complement* of the unions of groups. The focus of this paper is an efficient algorithm for the minimization of (2) with penalty function (4) and the corresponding selection of regularization parameter λ. However, the extension of this work to the penalty function of [38, 52] is also of interest.

2 Overlapping group shrinkage/thresholding

2.1 Motivation

As shown in [18, 43, 70], neighboring coefficients in a wavelet transform have statistical dependencies even when they are uncorrelated. In particular, a wavelet coefficient is more likely to be large in amplitude if the adjacent coefficients (in scale or spatially) are large. This behavior can be modeled using suitable non-Gaussian multivariate probability density functions, perhaps the simplest one being

$$p(\mathbf{x}) = \frac{C}{\sigma^d} \exp\left(-\frac{\sqrt{d+1}}{\sigma}\|\mathbf{x}\|_2\right), \quad \mathbf{x} \in \mathbb{C}^d \tag{5}$$

as utilized in [64]. If the coefficients $\mathbf{x}$ are observed in additive white Gaussian noise, $\mathbf{y} = \mathbf{x} + \mathbf{w}$, then the MAP estimator of $\mathbf{x}$ is obtained by solving (2) with $R(\mathbf{x}) = \|\mathbf{x}\|_2$, the solution of which is given by

$$\hat{\mathbf{x}} = \left(1 - \frac{\lambda}{\|\mathbf{y}\|}\right)_+ \mathbf{y}, \tag{6}$$

where $(x)_+ := \max(x, 0)$. The function (6) can be considered a multivariate form of soft thresholding with threshold λ.

The multivariate model and related models are convenient for estimating small blocks/neighborhoods within a large array of coefficients; however, when each coefficient is a member of multiple groups, then either estimated coefficients are discarded (e.g., only the center of each block of estimated coefficients is retained as in sliding window processing) or the blocks are sub-sampled so that they are not overlapping. In the first case, the result does not correspond directly to the minimization of a cost function; whereas, in the second case, the process is not translation-invariant and issues may arise due to blocks not aligning with the group sparsity structure within the array. Here we are interested in a variational approach based on fully-overlapping groups so that the derived algorithm is translation-invariant and blocking artifacts are avoided. The penalty function (4) is clearly translation-invariant.

2.2 Penalty function

Assuming that $\mathbf{x}$ has a group sparsity (clustering) property, to perform translation-invariant denoising, we add the ℓ_2 norm for each group to obtain the penalty function (4). The penalty function (4) is convex and the cost function $F : \mathbb{C}^N \to \mathbb{R}$ in (2), i.e.,

$$F(\mathbf{x}) = \frac{1}{2}\|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda \sum_{i \in \mathcal{I}} \left[\sum_{j \in \mathcal{J}} |x(i+j)|^2 \right]^{1/2}, \tag{7}$$

is strictly convex because $\|\mathbf{y} - \mathbf{x}\|_2^2$ is strictly convex in $\mathbf{x}$. The index i is the group index, and j is the coefficient index within group i. Each group has the same size, namely $|\mathcal{J}|$. In practice, $\mathbf{x}$ is of finite size; hence, the sum over i in (4) is a finite sum. To deal with the boundaries of $\mathbf{x}$, we take $x(i)$ in (4) as zero when i falls outside $\mathcal{I}$; that is, $x(i) = 0$ for $i \notin \mathcal{I}$.

For one-dimensional vectors $\mathbf{x}$ of length N with group size K, we set $\mathcal{I}$ in (1) to

$$\mathcal{I} = \{0, \ldots, N-1\}, \tag{8}$$

and $\mathcal{J}$ in (4) to

$$\mathcal{J} = \{0, \ldots, K-1\}. \tag{9}$$

Note that in (4), the groups are fully overlapping, as per a sliding window shifted a single sample at a time. It is possible to include a weighting $w(j)$ in (4) as in Ref. [68, 69].

For a two-dimensional array $\mathbf{x}$ of size $N_1 \times N_2$ with group size $K_1 \times K_2$, we set $\mathcal{I}$ in (1) to

$$\mathcal{I} = \{(i_1, i_2) : 0 \le i_1 \le N_1 - 1,\ 0 \le i_2 \le N_2 - 1\},$$

and $\mathcal{J}$ in (4) to

$$\mathcal{J} = \{(j_1, j_2) : 0 \le j_1 \le K_1 - 1,\ 0 \le j_2 \le K_2 - 1\}.$$

In the two-dimensional case, $i+j$ denotes (i_1+j_1, i_2+j_2). The same notation extends to higher dimensions.

Note that minimizing (2) with (4) can only shrink the data vector $\mathbf{y}$ toward zero. That is, the minimizer $\mathbf{x}^*$ of (2) will lie point-wise between zero and $\mathbf{y}$, i.e., $x^*(i) \in [0, y(i)]$ for all $i \in \mathcal{I}$. This can be shown by observing that the penalty function in (4) is a strictly increasing function of $|x(i)|$ and is independent of the sign of each $x(i)$. As a result, if $y(i) = 0$ for some $i \in \mathcal{I}$, then $x^*(i) = 0$ also.

An important point regarding R in (4) is that it is non-differentiable. In particular, if any group of $\mathbf{x}$ is equal to zero, then R is non-differentiable at $\mathbf{x}$.

2.3 Algorithm

To minimize the cost function $F(\mathbf{x})$ in (2), we use the majorization-minimization (MM) method [28]. To this end, we use the notation

$$\mathbf{v}_{i,K} = [v(i), \ldots, v(i+K-1)]^T \in \mathbb{C}^K \tag{10}$$

to denote the i-th group of vector $\mathbf{v}$. Then the penalty function in (4) can be written as

$$R(\mathbf{x}) = \sum_i \|\mathbf{x}_{i,K}\|_2. \tag{11}$$

To majorize $R(\mathbf{x})$, first note that

$$\frac{1}{2}\left[\frac{\|\mathbf{v}\|_2^2}{\|\mathbf{u}\|_2} + \|\mathbf{u}\|_2\right] \geq \|\mathbf{v}\|_2 \tag{12}$$

for all $\mathbf{v}$ and $\mathbf{u} \neq \mathbf{0}$, with equality when $\mathbf{v} = \mathbf{u}$. The inequality (12) can be derived from $t^2+s^2 \geq 2ts, \forall t, s \in \mathbb{R}$ by setting $t = \|\mathbf{v}\|_2$, $s = \|\mathbf{u}\|_2$. Define

$$g(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\sum_i \left[\frac{\|\mathbf{x}_{i,K}\|_2^2}{\|\mathbf{u}_{i,K}\|_2} + \|\mathbf{u}_{i,K}\|_2\right]. \tag{13}$$

If $\|\mathbf{u}_{i,K}\|_2 > 0$ for all $i \in \mathcal{I}$, then it follows from (11) and (12) that

$$g(\mathbf{x}, \mathbf{u}) \geq R(\mathbf{x}) \tag{14}$$

$$g(\mathbf{u}, \mathbf{u}) = R(\mathbf{u}), \tag{15}$$

for all $\mathbf{x}, \mathbf{u}$. Therefore, $g(\mathbf{x}, \mathbf{u})$ is a majorizer of $R(\mathbf{x})$ provided that $\mathbf{u}$ has no groups equal to zero. Moreover, the elements of $\mathbf{x}$ are decoupled in $g(\mathbf{x}, \mathbf{u})$ and so $g(\mathbf{x}, \mathbf{u})$ can be written as

$$g(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\sum_{i \in \mathcal{I}} r(i;\, \mathbf{u})|x(i)|^2 + c \tag{16}$$

where

$$r(i;\, \mathbf{u}) := \sum_{j \in \mathcal{J}} \left[\sum_{k \in \mathcal{J}} |u(i-j+k)|^2\right]^{-1/2} \tag{17}$$

and c is a constant that does not depend on $\mathbf{x}$. A majorizer of the cost function $F(\mathbf{x})$ is now given by

$$G(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda\, g(\mathbf{x}, \mathbf{u}). \tag{18}$$

The MM method produces the sequence $\mathbf{x}^{(k)}$, $k \geq 1$, given by:

$$\mathbf{x}^{(k+1)} = \arg\min_{\mathbf{x}} \; G(\mathbf{x}, \mathbf{x}^{(k)}) \tag{19}$$

$$= \arg\min_{\mathbf{x}} \; \|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda \sum_{i \in \mathcal{I}} r(i;\, \mathbf{x}^{(k)})\, |x(i)|^2 \tag{20}$$

where $\mathbf{x}^{(0)}$ is the initialization. Note that (20) is separable in $x(i)$, so we can write (20) equivalently as

$$x^{(k+1)}(i) = \arg\min_{x \in \mathbb{C}} \; |y(i) - x|^2 + \lambda\, r(i;\, \mathbf{x}^{(k)})\, |x|^2. \tag{21}$$

However, the term $r(i; \mathbf{x}^{(k)})$ is undefined if $\mathbf{x}_{i,K}^{(k)} = \mathbf{0}$, i.e., if the i-th group is all zero. This is a manifestation of the singularity issue which may arise whenever a quadratic function is used to majorize a non-differentiable function [28]. Hence, care must be taken to define an algorithm that avoids operations involving undefined quantities.

Consider the following algorithm. Define $\mathcal{I}'$ as the subset of $\mathcal{I}$ where $x^{(0)}(i) \neq 0$,

$$\mathcal{I}' := \{i \in \mathcal{I} \;:\; x^{(0)}(i) \neq 0\}. \tag{22}$$

Define the update equation by:

$$x^{(k+1)}(i) = \begin{cases} \dfrac{y(i)}{1 + \lambda\, r(i;\, \mathbf{x}^{(k)})}, & i \in \mathcal{I}' \\ 0 & i \notin \mathcal{I}' \end{cases} \tag{23}$$

with initialization $\mathbf{x}^{(0)} = \mathbf{y}$. Note that the first case of (23) is the solution to (21). The iteration (23) is the 'overlapping group shrinkage' (OGS) algorithm, summarized in Table 1 in the form of pseudo-code.

Several observations can be made.

It is clear from (22) and (23), that

$$x^{(0)}(i) = 0 \quad \Longrightarrow \quad x^{(k)}(i) = 0, \text{ for all } k \geq 1. \tag{24}$$

Therefore, any values initialized to zero will equal zero at every subsequent iteration of the algorithm. However, if $x^{(0)}(i) = 0$, then $y(i) = 0$ as per the initialization. Note that if $y(i) = 0$, then the optimal solution $\mathbf{x}^*$ has $x^*(i) = 0$. Therefore,

$$x^{(0)}(i) = 0 \quad \Longrightarrow \quad x^{(k)}(i) = x^*(i), \text{ for all } k \geq 1. \tag{25}$$

As a side note, if $\mathbf{y}$ is a noisy data vector, then it is unlikely that $y(i) = 0$ for any $i \in \mathcal{I}$.

Now consider the case where $x^{(0)}(i) \neq 0$. Note that if $x^{(k)}(i) \neq 0$, then no group of $\mathbf{x}^{(k)}$ containing $x^{(k)}(i)$ is entirely zero. Hence $r(i; \mathbf{x}^{(k)})$ is well defined with $r(i; \mathbf{x}^{(k)}) > 0$. Therefore $y(i)/[1 + \lambda\, r(i; \mathbf{x}^{(k)})]$ is well defined, lies strictly between zero and $y(i)$, and has the same sign as $y(i)$. Hence, by induction,

$$x^{(0)}(i) \neq 0 \quad \Longrightarrow \quad x^{(k)}(i) \neq 0, \text{ for all } k \geq 1. \tag{26}$$

Therefore, any value not initialized to zero will never equal zero in any subsequent iteration. However, for

Table 1: Overlapping group shrinkage (OGS) algorithm for minimizing (2) with penalty function (4).

Algorithm OGS

input: $\mathbf{y} \in \mathbb{C}^N$, $\lambda > 0$, $\mathcal{J}$

$\mathbf{x} = \mathbf{y}$ (initialization)

$\mathcal{I}' = \{i \in \mathcal{I},\ x(i) \neq 0\}$

repeat

$$r(i) = \sum_{j \in \mathcal{J}} \left[\sum_{k \in \mathcal{J}} |x(i-j+k)|^2 \right]^{-1/2}, \quad i \in \mathcal{I}'$$

$$x(i) = \frac{y(i)}{1 + \lambda\, r(i)}, \quad i \in \mathcal{I}'$$

until convergence

return: $\mathbf{x}$

---

some $i \in \mathcal{I}'$, $x^{(k)}(i)$ may approach zero in the limit as $k \to \infty$. That is,

$$x^{(0)}(i) \neq 0 \quad \not\Longrightarrow \quad \lim_{k \to \infty} x^{(k)}(i) \neq 0. \tag{27}$$

In the example in Sect. 4.1, it will be seen that some values do go to zero as the algorithm progresses. In practice, (26) may fail for some $i \in \mathcal{I}'$ due to finite numerical precision, i.e., some $x(i)$ may equal zero at some iteration even if it was not initialized to zero. In this case, such i should be removed from $\mathcal{I}'$ to avoid potential 'division-by-zero' in subsequent iterations.

The OGS algorithm produces sparse solutions by gradually reducing non-zero values of $\mathbf{y}$ toward zero, rather than by thresholding them directly to zero on any iteration, as illustrated in Fig. 4 below.

The output of the OGS algorithm will be denoted as $\mathbf{x} = \mathrm{ogs}(\mathbf{y};\, \lambda, K)$, where K is the block size. The OGS algorithm can also be applied to multidimensional data $\mathbf{y}$ using the above multi-index notation, with group size $K = (K_1, \ldots, K_d)$ and where $\mathcal{I}$ and $\mathcal{J}$ are multi-indexed sets as described above.

Note that when the group size is $K = 1$, then $\mathcal{J}$ in (9) is given by $\mathcal{J} = \{0\}$, and the penalty function (4) is simply the ℓ_1 norm of $\mathbf{x}$. In this case, the solution $\mathbf{x}^*$ can be obtained exactly and non-iteratively by soft thresholding. When the group size K is greater than one, the groups overlap and every element of the solution $\mathbf{x}^*$ depends on every element of $\mathbf{y}$; hence, it is not possible to display a multivariate shrinkage function as in [64] for the non-overlapping case.

Implementation. The quantity $r(i;\, \mathbf{x})$ in step (2) of the OGS algorithm can be computed efficiently using two convolutions — one for the inner sum and one for the outer sum. The inner sum can be computed as the convolution of $|x(\cdot)|^2$ with a 'boxcar' of size $|\mathcal{J}|$. Denoting by $g(\cdot)$ the result of the inner sum, the outer sum is again a running sum or 'boxcar' convolution applied to $g(\cdot)^{-1/2}$. In the multidimensional case, each of the two convolutions are multidimensional but separable and hence computationally efficient.

The computational complexity of each iteration of the OGS algorithm is of order KN, where K is the group size and N is the total size of $\mathbf{y}$. The memory required for the algorithm is $2N + K$.

Convergence. For the OGS algorithm, due to its derivation using majorization-minimization (MM), it

is guaranteed that the cost function $F(\mathbf{x})$ monotonically decreases from one iteration to the next, i.e., $F(\mathbf{x}^{(k+1)}) < F(\mathbf{x}^{(k)})$ if $\mathbf{x}^{(k)} \neq \mathbf{x}^*$. Yet, the proof of its convergence to the unique minimizer is complicated by the 'singularity issue' which arises when a quadratic function is used as a majorizer of a non-differentiable function [28]. For the OGS problem it is particularly relevant since, as in most sparsity-based denoising problems, $F(\mathbf{x})$ will usually be non-differentiable at the minimizer $\mathbf{x}^*$. Specifically, for OGS, if $\mathbf{x}^*$ contains any group that is all zero, then $F(\mathbf{x})$ is non-differentiable at $\mathbf{x}^*$. However, several results regarding the singularity issue are given in [28] which strongly suggest that this issue need not hinder the reliable convergence of such algorithms in practice.

For the OGS algorithm with the initialization $\mathbf{x}^{(0)} = \mathbf{y}$, the component $\mathbf{x}^{(k)}(i)$ will never equal zero except when $y(i)$ itself equals zero, as noted above. In the OGS algorithm, the singularity issue is manifested by $r(i, \mathbf{x}^{(k)})$ approaching infinity for some $i \in \mathcal{I}'$. In particular, if $x^*(i) = 0$ for some $i \in \mathcal{I}'$, then $r(i, \mathbf{x}^*)$ is undefined (due to 'division-by-zero'). For $i \in \mathcal{I}'$ such that $x^*(i) = 0$, the value of $r(i, \mathbf{x}^{(k)})$ goes to infinity as the algorithm progresses, and $x^{(k)}(i)$ goes to zero. Note that in the OGS algorithm, the term $r(i, \mathbf{x}^{(k)})$ is used only in the expression $y(i)/[1 + \lambda\, r(i; \mathbf{x}^{(k)})]$. Therefore, even when $r(i; \mathbf{x}^{(k)})$ is very large, this expression is still numerically well behaved. (When large values of opposite signs are added to obtain small numbers, the result is not numerically reliable, but that is not the case here.) If the algorithm is implemented in fixed point arithmetic, then it is indeed likely that the large values of $r(i; \mathbf{x}^{(k)})$ will lead to overflow for some $i \in \mathcal{I}'$. In this case, $x^{(k)}(i)$ should be set to zero and i should be removed from $\mathcal{I}'$ for the subsequent iterations.

We do not prove the convergence of the OGS algorithm due to the singularity issue. This is illustrated in the Appendix. However, in practice, the singularity issue does not appear to impede the convergence of the algorithm, consistent with the results of [28]. We have found that the empirical asymptotic convergence behavior compares favorably with other algorithms that have proven convergence, as illustrated in Fig. 5.

One approach to avoid the singularity issue is to use the differentiable penalty function

$$R_\epsilon(\mathbf{x}) = \sum_{i \in \mathcal{I}} \left[\left(\sum_{j \in \mathcal{J}} |x(i+j)|^2 \right) + \epsilon \right]^{\frac{1}{2}} \tag{28}$$

where ϵ is a small positive value, instead of (4). However, as in [53], we have found it unnecessary to do so, since we have found that the algorithm is not hindered by the singularity issue in practice. For the regularizer (28), convergence of the corresponding form of OGS can be proven using the Global Convergence Theorem (GCT) [45, 60].

Proximal operator. An effective approach for solving a wide variety of inverse problems is given by the proximal framework [4,15,16]. In this approach, the solution $\mathbf{x}$ to a general inverse problem (e.g. deconvolution, interpolation, reconstruction) with penalty function $R(\mathbf{x})$ can be computed by solving a sequence of denoising problems each with penalty function $R(\mathbf{x})$. Therefore, the efficient computation of the solution to the denoising problem is important for the use of proximal methods. In this context, the denoising problem, i.e., (2), is termed the *proximal operator* (or proximity operator). The OGS algorithm is therefore an implementation of the proximal operator for penalty function (4), and can be used in proximal algorithms for inverse problems that are more general than denoising. As noted above, other implementations of the proximal operator associated with overlapping group sparsity have been given [2, 3, 7, 12, 20, 27, 38, 40, 54, 74]; these algorithms are based on the duplication of variables, and hence require more memory (proportional to the group size in the fully-overlapping case).

FOCUSS. The OGS algorithm can be considered as a type of FOCUSS algorithm [34] that is designed to yield sparse solutions to under-determined linear systems of equations. It was extended to more general

penalty functions (or diversity measures) [60] and to the noisy-data case [59]. Setting $p = 1$ and $A = \mathbf{I}$ in equation (13) of [59] gives the OGS algorithm for group size $K = 1$, namely an iterative implementation of the soft-threshold rule. (Note, the emphasis of FOCUSS is largely on the non-convex ($p < 1$) case with $A \neq \mathbf{I}$ and group size $K = 1$, i.e., the non-group case.)

The FOCUSS algorithm was further extended to the case of multiple measurement vectors (MMV) that share a common sparsity pattern [17]. We note that the resulting algorithm, M-FOCUSS, is different than OGS. In the MMV model, the location (indices) of significant coefficients is consistent among a set of vectors; while in the OGS problem there is no common (repeated) sparsity pattern to be exploited.

The M-FOCUSS algorithm was later improved to account for gradual changes in the sparsity pattern in a sequence of measurement vectors [76]. This approach involves, in part, arranging the sequence of measurement vectors into overlapping blocks. While both, the algorithm of [76] and the OGS algorithm, utilize overlapping blocks, the former algorithm utilizes overlapping blocks to exploit a sparsity pattern (approximately) common to multiple measurement vectors, whereas OGS does not assume any common sparsity among blocks.

The FOCUSS algorithm was extended to mixed norms in [41] to attain structured sparsity without overlapping groups. This approach is extended in [42, 68, 69] where sliding windows are used to account for overlapping groups. However, as noted in [42], this approach does not directly correspond to an optimization problem; hence it does not constitute an implementation of the proximal operator.

3 Gaussian noise and OGS

This section addresses the problem of how to set the regularization parameter λ in (2) in a simple and direct way analogous to the '3σ rule'. The use of the 3σ rule for soft thresholding, as illustrated in Sec. 4.1, is simple to apply because soft thresholding has a simple explicit form. However, overlapping group shrinkage has no explicit form. Therefore, extending the notion of the '3σ rule' to OGS is not straight forward. The question addressed in the following is: how should λ be chosen so that essentially all the noise is eliminated? In principle, the minimum such λ should be used, because a larger value will only cause further signal distortion.

In order to set λ so as to reduce Gaussian noise to a desired level, the effect of the OGS algorithm on pure i.i.d. zero-mean unit-variance Gaussian noise is investigated. We examine first the case of group size $K = 1$, because analytic formulas can be obtained in this case (OGS being soft thresholding for $K = 1$).

Let $y \sim \mathcal{N}(0, 1)$ and define $x = \mathrm{soft}(y, T)$. Then the variance of x as a function of threshold T is given by

$$\sigma_x^2(T) = E[x^2] = \int_{|y|>T} (|y| - T)^2 \, p_y(y) \, dy \tag{29}$$

$$= 2\,(1 + T^2)\,Q(T) - T\sqrt{\frac{2}{\pi}} \exp\left(-\frac{T^2}{2}\right) \tag{30}$$

where $p_y(y)$ is the standard normal pdf $\mathcal{N}(0, 1)$ and

$$Q(T) := \frac{1}{\sqrt{2\,\pi}} \int_T^\infty \mathrm{e}^{-\frac{t^2}{2}} \, dt = 0.5 \left(1 - \mathrm{erf}\Big(\frac{T}{\sqrt{2}}\Big)\right).$$

The standard deviation $\sigma_x(T)$ is illustrated in Fig. 1a as a function of threshold T. Since the variance of x is unity here, the 3σ rule suggests setting the threshold to $T = 3$ which leads to $\sigma_x(3) = 0.020$ according to (30).

The graph in Fig. 1a generalizes the 3σ rule: Given a specified output standard deviation σ_x, the graph shows how to set the threshold T in the soft threshold function so as to achieve it, i.e., so that $E[\mathrm{soft}(y, T)^2] =$

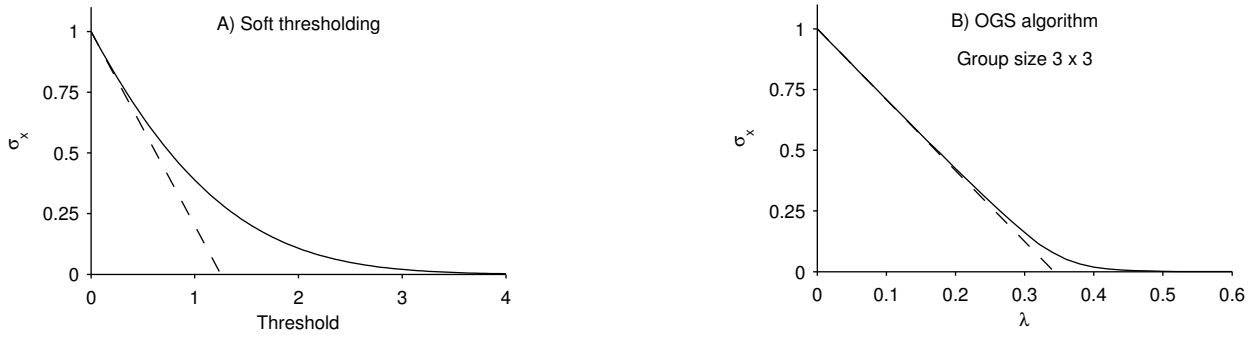

Figure 1: Standard deviation of standard Gaussian noise $\mathcal{N}(0,1)$ after (a) soft thresholding and (b) overlapping group shrinkage (OGS) with group size 3×3.

σ_x^2 where $y \sim \mathcal{N}(0,1)$. For example, to reduce the noise standard deviation σ to one percent of its value, we solve $\sigma_x(T) = 0.01$ for T to obtain $T = 3.36\sigma$. This threshold is greater than that suggested by the 3σ rule.

To set the regularization parameter λ in the OGS algorithm, we suggest that the same procedure can be followed. However, for OGS there is no explicit formula such as (30) relating λ to σ_x. Indeed, in the overlapping group case, neither is it possible to reduce $E[x^2]$ to a univariate integral as in (29) due to the coupling among the components of $\mathbf{y}$, nor is there an explicit formula for $\mathbf{x}$ in terms of $\mathbf{y}$, but only a numerical algorithm.

Although no explicit analog of (30) is available for OGS, the functional relationship can be found numerically. Let $\mathbf{y}$ be i.i.d. $\mathcal{N}(0,1)$ and define $\mathbf{x}$ as the output of the OGS algorithm: $\mathbf{x} = \text{ogs}(\mathbf{y};\, \lambda,\, K)$. The output standard deviation σ_x can be found by simulation as a function of λ for a fixed group size. For example, consider applying the OGS algorithm to a two-dimensional array $\mathbf{y}$ using a group size of 3×3. For this group size, σ_x as a function of λ is illustrated in Fig. 1b. The graph is obtained by generating a large two-dimensional array of i.i.d. standard normal random variables, applying the OGS algorithm for a discrete set of λ, and then computing the standard deviation of the result for each λ. Once this graph is numerically obtained, it provides a straight forward way to set λ so as to reduce the noise to a specified level. For example, to reduce the noise standard deviation σ down to one percent of its value, we should use $\lambda \approx 0.43\sigma$ in the OGS algorithm according to the graph in Fig. 1b. (Obtaining the value λ corresponding to a specified σ_x generally requires an extrapolation between the data points comprising a graph such as Fig. 1b.) Note that the graph will depend on the group size, so for a different group size the graph (or table) needs to be recalculated. For efficiency, these calculations can be performed off-line for a variety of group sizes and stored for later use.

Table 2 gives the value of the regularization parameter λ so that OGS produces an output signal $\mathbf{x}$ with specified standard deviation σ_x when the input $\mathbf{y}$ is standard normal (zero-mean unit-variance i.i.d. Gaussian). The table applies to both 1D and 2D signals. The table provides values for groups up to length 5 in 1D, and up to size 5×5 in 2D. Note for groups of size $1 \times K$, the value of λ is the same for 1D and 2D. Also, note that λ is the same for groups of size $K_1 \times K_2$ and $K_2 \times K_1$; so each is listed once in the table. The first value in each entry is obtained using 150 iterations of the OGS algorithm (more than sufficient for accurate convergence); while the second value in parenthesis is obtained using 25 iterations (sufficient in practice and faster to compute). For group size 1×1, OGS is simply soft thresholding; hence no iterations are needed for it. The values λ listed in Table 2 are accurate to within 0.01 and were computed via simulation.

To clarify how Table 2 is intended to be used, suppose one is using the OGS algorithm with 2×3 groups for denoising a signal contaminated by additive white Gaussian noise with standard deviation σ. To reduce the noise standard deviation down to 0.1% of its value, one would use $\lambda = 0.74\sigma$ if running the OGS algorithm

Table 2: Parameter λ for standard normal i.i.d. signal

	Output standard deviation, σ_x			
Group	10^{-2}	10^{-3}	10^{-4}	10^{-5}
1×1	3.36	4.38	5.24	6.00
1×2	1.69 (1.73)	2.15 (2.24)	2.38 (2.61)	2.46 (2.94)
1×3	1.16 (1.18)	1.46 (1.52)	1.60 (1.77)	1.64 (1.99)
1×4	0.89 (0.91)	1.12 (1.16)	1.23 (1.36)	1.27 (1.53)
1×5	0.73 (0.75)	0.92 (0.95)	1.01 (1.12)	1.04 (1.25)
2×2	0.86 (0.87)	1.08 (1.13)	1.19 (1.31)	1.23 (1.48)
2×3	0.59 (0.61)	0.74 (0.77)	0.80 (0.89)	0.82 (1.01)
2×4	0.46 (0.48)	0.57 (0.59)	0.62 (0.69)	0.64 (0.78)
2×5	0.38 (0.41)	0.46 (0.49)	0.51 (0.57)	0.52 (0.64)
3×3	0.41 (0.43)	0.50 (0.53)	0.55 (0.61)	0.56 (0.69)
3×4	0.33 (0.35)	0.39 (0.42)	0.43 (0.48)	0.44 (0.54)
3×5	0.29 (0.31)	0.32 (0.36)	0.35 (0.40)	0.36 (0.45)
4×4	0.27 (0.30)	0.30 (0.34)	0.33 (0.38)	0.34 (0.43)
4×5	0.24 (0.26)	0.26 (0.30)	0.27 (0.33)	0.28 (0.37)
5×5	0.21 (0.23)	0.22 (0.26)	0.23 (0.29)	0.24 (0.32)
2×8	0.28 (0.30)	0.31 (0.35)	0.33 (0.39)	0.35 (0.43)

Regularization parameter λ to achieve specified output standard deviation when OGS is applied to a real standard normal signal: full convergence (25 iterations).

to full convergence; or one would use $\lambda = 0.77\sigma$ if using only 25 iterations. These values are from the 10^{-3} column in Table 2.

It can be observed in Fig. 1 that the function $\sigma_x(\cdot)$ has a sharper 'knee' in the case of OGS compared with soft thresholding. We have examined graphs for numerous group sizes and found that in general the larger the group, the sharper is the knee. Note that in practice λ should be chosen large enough to reduce the noise to a sufficiently negligible level, yet no larger so as to avoid unnecessary signal distortion. That is, suitable values of λ are somewhat near the knee. Therefore, due to the sharper knee, the denoising process is more sensitive to λ for larger group sizes; hence, the choice of λ is more critical.

Similarly, it can be observed in Fig. 1 that for OGS, the function $\sigma_x(\cdot)$ follows a linear approximation more closely to the left of the 'knee' than it does in the case of soft thresholding. We have found that near the origin, $\sigma_x(\lambda)$ is approximated by

$$\sigma_x(\lambda) \approx -\sqrt{2}\,\frac{\Gamma(|\mathcal{J}|/2 + 1/2)}{\Gamma(|\mathcal{J}|/2)}\,\lambda, \quad \text{for } \lambda \approx 0, \tag{31}$$

where $|\mathcal{J}|$ is the cardinality of the group (K in 1D, K_1K_2 in 2D). This can be explained by noting that for $\mathbf{y} \sim \mathcal{N}(0, \sigma^2)$, the ℓ_2 norm of the group follows a chi-distribution with $|\mathcal{J}|$ degrees of freedom, the mean of which is the slope in (31). For small λ, OGS has roughly the effect of soft thresholding the ℓ_2 norm of the groups. However, it should be noted that small λ are of minor interest in practice because noise will not be sufficiently attenuated. The right hand side of (31) is illustrated as dashed lines in Fig. 1.

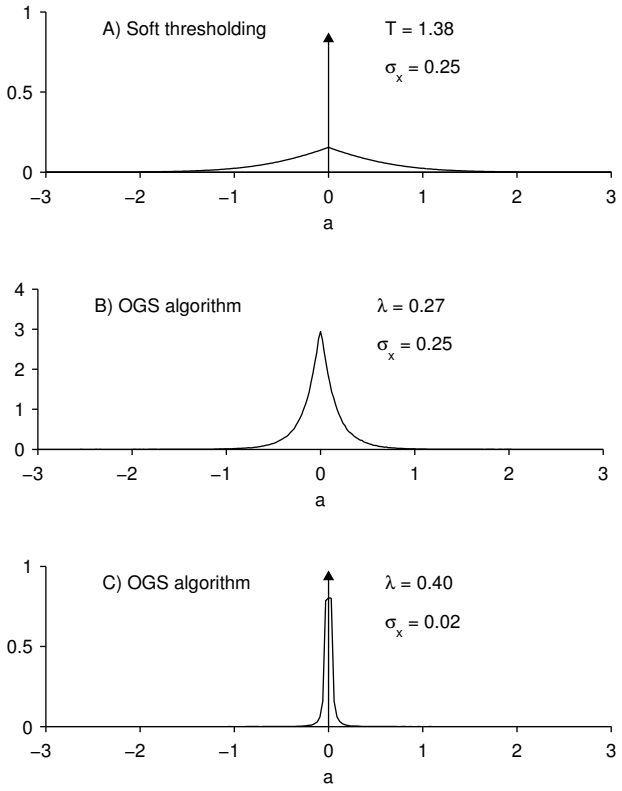

Figure 2: Probability density function of zero-mean unit-variance Gaussian noise after soft thresholding (a) and OGS (b,c). In (a) and (b), the parameters (T and λ) are set so that $\sigma_x = 0.25$. In (b) and (c) OGS was applied with group size 3×3. In (a) and (c) the pdf contains a point mass at zero.

3.1 Shrinkage/Thresholding behavior

Although not apparent in Fig. 1, the soft-thresholding and OGS procedures are quite distinct in their shrinkage/thresholding behavior. Clearly, if $\mathbf{y}$ is i.i.d. with $y \sim \mathcal{N}(0,1)$ and if $\mathbf{x}$ is the result of soft thresholding (i.e., $x = \text{soft}(y,T)$), then $\mathbf{x}$ contains many zeros. All values $|y| \leq T$ are mapped to zero. The nonlinearity involves *thresholding* in this sense. In contrast, OGS does not produce any zeros unless λ is sufficiently large. This behavior is illustrated in Fig. 2, which shows the pdf of y after soft thresholding (a) and after OGS (b, c). Soft thresholding with $T = 1.38$ produces an output x with $\sigma_x = 0.25$. The pdf of x, illustrated in Fig. 2a, consists of a point mass at the origin of mass 0.831 and the tails of the Gaussian pdf translated toward the origin. The point mass represents the zero elements of x.

Using OGS with group size 3×3 and λ set so that again $\sigma_x = 0.25$, the output x does not contain zeros. The pdf is illustrated in Fig. 2b. The absence of the point mass at the origin reflects the fact that OGS mapped no values of y to zero, i.e., no thresholding. The pdf in Fig. 2b is computed numerically by applying the OGS algorithm to simulated standard normal data, as no explicit formula is available.

When λ is sufficiently large, then OGS does perform thresholding, as illustrated in Fig. 2c. For a group size of 3×3 and $\lambda = 0.4$, the pdf exhibits a point-mass at the origin reflecting the many zeros in the output of OGS.[1] For this value of λ, OGS performs both thresholding and shrinkage, like the soft threshold function.

[1]The pdf in Fig. 2c is computed as a histogram; hence, the exact value of the point-mass at the origin depends on the histogram bin width.

Table 3: Parameter λ for standard complex normal i.i.d. signal

	Output standard deviation, σ_x			
Group	10^{-2}	10^{-3}	10^{-4}	10^{-5}
1×1	2.54	3.26	3.86	4.39
1×2	1.30 (1.33)	1.65 (1.71)	1.82 (2.00)	1.89 (2.25)
1×3	0.90 (0.93)	1.12 (1.17)	1.22 (1.35)	1.26 (1.52)
1×4	0.71 (0.73)	0.86 (0.91)	0.93 (1.04)	0.96 (1.17)
1×5	0.60 (0.62)	0.71 (0.75)	0.76 (0.86)	0.78 (0.96)
2×2	0.66 (0.69)	0.83 (0.87)	0.91 (1.01)	0.94 (1.13)
2×3	0.48 (0.51)	0.56 (0.60)	0.61 (0.69)	0.63 (0.78)
2×4	0.39 (0.42)	0.44 (0.49)	0.47 (0.55)	0.49 (0.61)
2×5	0.34 (0.37)	0.37 (0.42)	0.39 (0.47)	0.41 (0.53)
3×3	0.36 (0.39)	0.40 (0.45)	0.42 (0.50)	0.43 (0.56)
3×4	0.30 (0.33)	0.32 (0.38)	0.34 (0.42)	0.35 (0.47)
3×5	0.27 (0.29)	0.28 (0.33)	0.29 (0.37)	0.30 (0.41)
4×4	0.26 (0.28)	0.27 (0.32)	0.28 (0.36)	0.29 (0.40)
4×5	0.23 (0.25)	0.24 (0.28)	0.25 (0.32)	0.25 (0.35)
5×5	0.20 (0.22)	0.21 (0.25)	0.22 (0.28)	0.22 (0.31)
2×8	0.26 (0.28)	0.27 (0.32)	0.28 (0.36)	0.29 (0.40)

3.2 Complex data

The calculation (30) is for real-valued standard normal x. For complex-valued Gaussian data the formula is slightly different:

$$\sigma_x^2 = \int_{|y|>T} ||y| - T|^2 \, p_y(y) \, dy \tag{32}$$

$$= \exp(-T^2) - 2\sqrt{\pi}\, T \, Q(\sqrt{2}\, T) \tag{33}$$

where $p_y(y)$ is the zero-mean unit-variance complex Gaussian pdf (standard complex normal), $\mathcal{CN}(0, 1)$. In the complex case, (31) is modified to

$$\sigma_x(\lambda) \approx -\frac{\Gamma(|\mathcal{J}| + 1/2)}{\Gamma(|\mathcal{J}|)} \, \lambda, \quad \text{for } \lambda \approx 0, \tag{34}$$

as the degrees of freedom of the chi-distribution is doubled (due to real and imaginary parts of complex y). Because complex-valued data is common (using the Fourier transform, STFT, and in radar and medical imaging, etc.), it is useful to address the selection of λ in the complex case. Note that Table 2 does not apply to the complex case. Table 3 gives the value of λ for the complex case, analogous to Table 2 for the real case.

3.3 Remarks

The preceding sections described how the parameter λ may be chosen so as to reduce additive white Gaussian noise to a desired level. However, in many cases the noise is not white. For example, in the speech denoising example in Section 4.2, the OGS algorithm is applied directly in the STFT domain. However, the STFT is an overcomplete transform; therefore, the noise in the STFT domain will not be white, even if it is white in the original signal domain. In the speech denoising example in Section 4.2 below, this issue is ignored and the algorithm is still effective. However, in other cases, where the noise is more highly correlated, the values

λ in Table 2 and 3 may be somewhat inaccurate.

The penalty function (4) is suitable for stationary noise; however, in many applications, noise is not stationary. For example, in the problem of denoising speech corrupted by stationary colored noise, the variance of the noise in the STFT domain will vary as a function of frequency. In particular, some noise components may be narrowband and therefore occupy a narrow time-frequency region. The OGS penalty function and algorithm, as described in this paper, do not apply to this problem directly. The penalty function (4) and the process to select λ must be appropriately modified.

The OGS algorithm as described above uses the same block size over the entire signal. In some applications, it may be more appropriate that the block size varies. For example, in speech denoising, as noted and developed in [73], it is beneficial that the block size in the STFT domain varies as a function of frequency (e.g., for higher temporal resolution at higher frequency). In addition, the generalization of OGS so as to denoise wavelet coefficients on tree-structured models as in [61] may be of interest.

4 Examples

4.1 One-dimensional denoising

As an illustrative example, the OGS algorithm is applied to denoising the one-dimensional group sparse signal in Fig. 3a. The noisy signal in Fig. 3b is obtained by adding independent white Gaussian noise with standard deviation $\sigma = 0.5$. The dashed line in the figure indicates the '3σ' level. The '3σ rule' states that nearly all values of a Gaussian random variable lie within three standard deviations of the mean (in fact, 99.7%). Hence, by using 3σ as a threshold with the soft threshold function, nearly all the noise will be eliminated as illustrated in Figure 3c with threshold $T = 3\sigma = 1.5$. Although the noise is effectively eliminated, the large-amplitude values of the signal have been attenuated, which is unavoidable when applying soft thresholding.

Even though this choice of threshold value does not minimize the RMSE, it is a simple and intuitive procedure which can be effective and practical in certain applications. Moreover, this rule does not depend on the signal energy (only the noise variance), so it is straight-forward to apply. Regarding the RMSE, it should be noted that optimizing the RMSE does not always lead to the most favorable denoising result in practice. For example, in speech enhancement/denoising, even a small amount of residual noise will be audible as 'musical noise' [44]. (In speech denoising via STFT thresholding, the RMSE-optimal threshold produces a denoised signal of low perceptual quality.)

The result of applying the OGS algorithm to the noisy signal is illustrated in Fig. 3d. Twenty-five iterations of the algorithm were used, with group size $K = 5$ and parameter[2] $\lambda = 0.68\sigma = 0.34$. As visible, the noise has been essentially eliminated. Compared to soft thresholding in Fig. 3c, the large-amplitude values of the original signal are less attenuated, and hence the RMSE is improved (0.52 compared to 0.82). In this example, the RMSE comparison between the two methods is meaningful because both methods have been used with parameter values chosen so as to eliminate essentially all the noise.

The convergence of $\mathbf{x}^{(k)}$ to $\mathbf{x}^*$ is illustrated in Fig. 4, where it can be seen that numerous values $x^{(k)}(i)$ converge to zero. In this example, with initialization $x^{(0)}(i) \neq 0$, $\forall i \in \mathcal{I}$, we have $x^{(k)}(i) \neq 0$, $\forall i \in \mathcal{I}$ for all subsequent iterations $k \geq 1$. Yet, entire groups converge to zero.

The convergence behavior in terms of the cost function history of the OGS algorithm is compared with three other algorithms in Fig. 5, namely to ADMM [20], CDAD (coordinate descent algorithm for the dual

[2]The parameter λ is chosen so that 25 iterations of the OGS algorithm with group size $K = 5$ applied to white Gaussian noise produces the same output standard deviation as soft thresholding using threshold $T = 3\sigma$. This method for selecting λ is elaborated in Section 3.

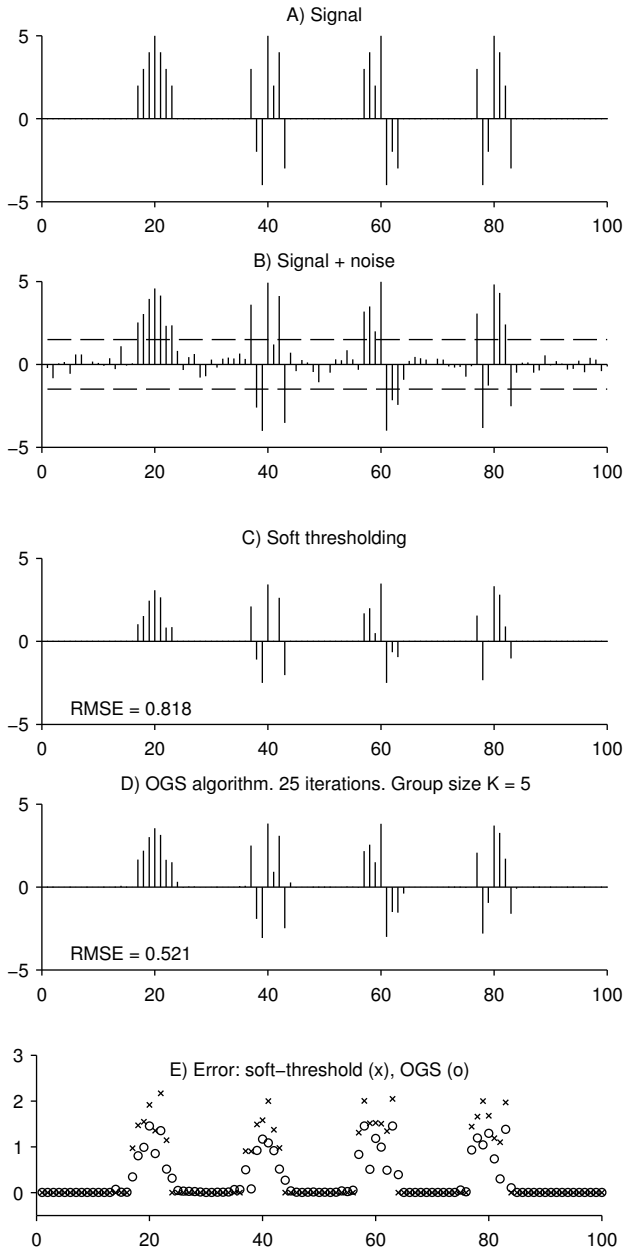

Figure 3: Group sparse signal denoising: comparison of soft thresholding and overlapping group shrinkage (OGS). OGS yields the smaller RMSE.

problem) [3, 40], and the algorithm of [74] as implemented in the SLEP (Sparse Learning with Efficient Projections) software package.[3] The figure shows the cost function history for the first 150 iterations of each algorithm. The OGS algorithm exhibits a monotone decreasing cost function that attains the lowest cost function value after 25 iterations. The ADMM, CDAD, and OGS algorithms have about the same computational cost per iteration, $O(NK)$. We implemented these three algorithms in MATLAB. The SLEP software is written in C (compiled in MATLAB as a mex file) and each iteration of SLEP performs several

[3]The SLEP software was downloaded from http://www.public.asu.edu/~jye02/Software/SLEP/. We thank the author for verifying the correctness of our usage of SLEP. We also note that SLEP is a versatile suite of algorithms that can solve a variety of sparsity-related optimization problems, not just the one considered here.

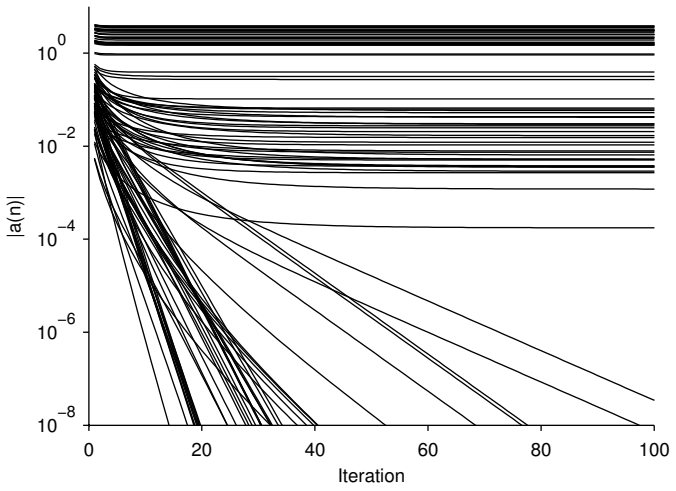

Figure 4: Convergence of OGS algorithm as applied in Fig. 3 .

inner optimization iterations.

The run-time for 150 iterations of SLEP, CDAD, ADMM, and OGS are 3, 99.5, 36.7 and 7.7 milliseconds, respectively. Note that SLEP and OGS are much faster than the other two. SLEP is written in C/Mex, so it is difficult to compare its run-time with the other three algorithms which are written in MATLAB. The MATLAB implementation of OGS is fast due to (*i*) its minimal data indexing, (*ii*) its minimal memory usage (no auxiliary/splitting variables), (*iii*) its computational work is dominated by convolution which is implemented very efficiently with the MATLAB built-in `conv` function.

It should also be noted that ADMM requires that parameters be specified by the user, which we manually set so as to optimize its convergence behavior. The algorithm of [74] also calls for parameters, but the SLEP software provides default values which we used here. The CDAD and OGS algorithms do not require any user parameters to be specified. We also note that ADMM and CDAD require 5-times the memory of OGS, as the group size is $K = 5$ and the groups are fully overlapping. In summary, the OGS algorithm has minimal memory requirements, requires no user specified parameters, and has the most favorable asymptotic convergence behavior.

We have found that for some problems OGS has a slow convergence during the initial iterations, compared to the other algorithms. This is because OGS does not perform explicit thresholding as do the other algorithms; instead, OGS gradually reduces values toward zero. It may be appropriate to perform several iterations of another algorithm or preliminary thresholding to improve initial convergence, and then switch to OGS. The comparison here uses OGS alone.

Partially overlapping groups. It may be expected that partially overlapping groups may also serve as a useful signal model. To this end, we performed numerical experiments to evaluate the utility of partially overlapping group sparsity by modifying the penalty function (4) so that the outer sum is over a sub-grid of $\mathcal{I}$. We used a set of signals like that of Fig. 3a where each group is systematically translated in turn, and we modified the OGS algorithm accordingly. For each signal, the RMSE-optimal λ was numerically found, and the corresponding optimal RMSE recorded for several values of the noise std, σ. Averaging over 100 realizations for each group position and overlap, we obtain the RMSE values shown in Table 4.1. The fully-overlapping case (overlap $M = 4$) gives the lowest RMSE, as might be expected. However, the non-overlapping case (overlap $M = 0$) does not give the worst RMSE. It turns out that partial overlapping can yield an inferior RMSE compared to both fully-overlapping and non-overlapping cases. The reason for this is

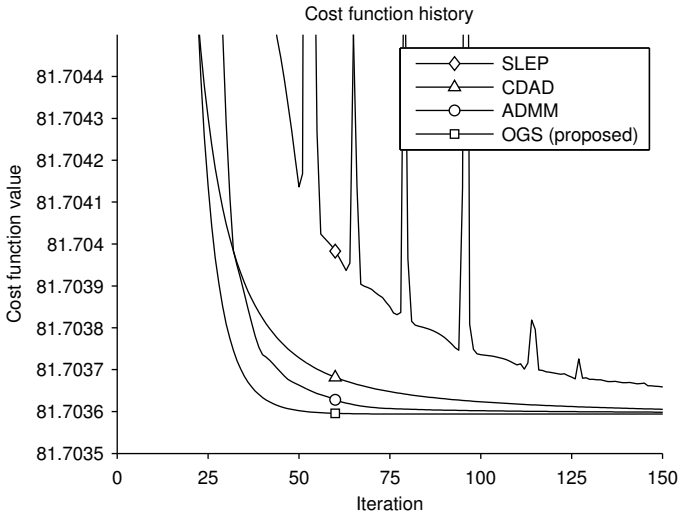

Figure 5: Cost function history: comparison of algorithms for 1-D denoising.

Table 4: Average minimum RMSE for partial overlap ($K = 5$)

	overlap, M				
σ	0	1	2	3	4
0.5	0.3925	0.3978	0.3935	0.3886	0.3846
1	0.7282	0.7391	0.7344	0.7248	0.7176
2	1.2135	1.2303	1.2291	1.2080	1.1954

that, in the partial overlapping case, some components of $\mathbf{x}$ may be a member of only one group, while other components will be a member of two or more groups. The more groups a component is a member of, the more strongly it is penalized. Hence, components of $\mathbf{x}$ are non-uniformly penalized in the partially-overlapping case, and this degrades its performance when the group structure is not known *a priori*.

4.2 Speech Denoising

This section illustrates the application of overlapping group shrinkage (OGS) to the problem of speech enhancement (denoising) [44]. The short-time Fourier transform (STFT) is the basis of many algorithms for speech enhancement, including classical spectrum subtraction [6, 48], improved variations thereof using non-Gaussian models [35, 47], and methods based on Markov models [21, 26, 72]. A well known problem arising in many speech enhancement algorithms is that the residual noise is audible as 'musical noise' [5]. Musical noise may be attributed to isolated noise peaks in the time-frequency domain that remain after processing. Methods to reduce musical noise include over estimating the noise variance, imposing a minimum spectral noise floor [5], and improving the estimation of model parameters [9]. Due to the musical noise phenomenon, it is desirable to reduce the noise sufficiently, even if doing so is not optimal with respect to the RMSE.

To avoid isolated spurious time-frequency noise spikes (to avoid musical noise), the grouping/clustering behavior of STFT coefficients of speech waveforms can be taken into account. To this end, a recent algorithm by Yu et al. [73] for speech/audio enhancement consists of time-frequency block thresholding. We note that the algorithm of [73] is based on non-overlapping blocks. Like the algorithm of [73], the OGS algorithm aims to draw on the grouping behavior of STFT coefficients so as to improve the overall denoising result,

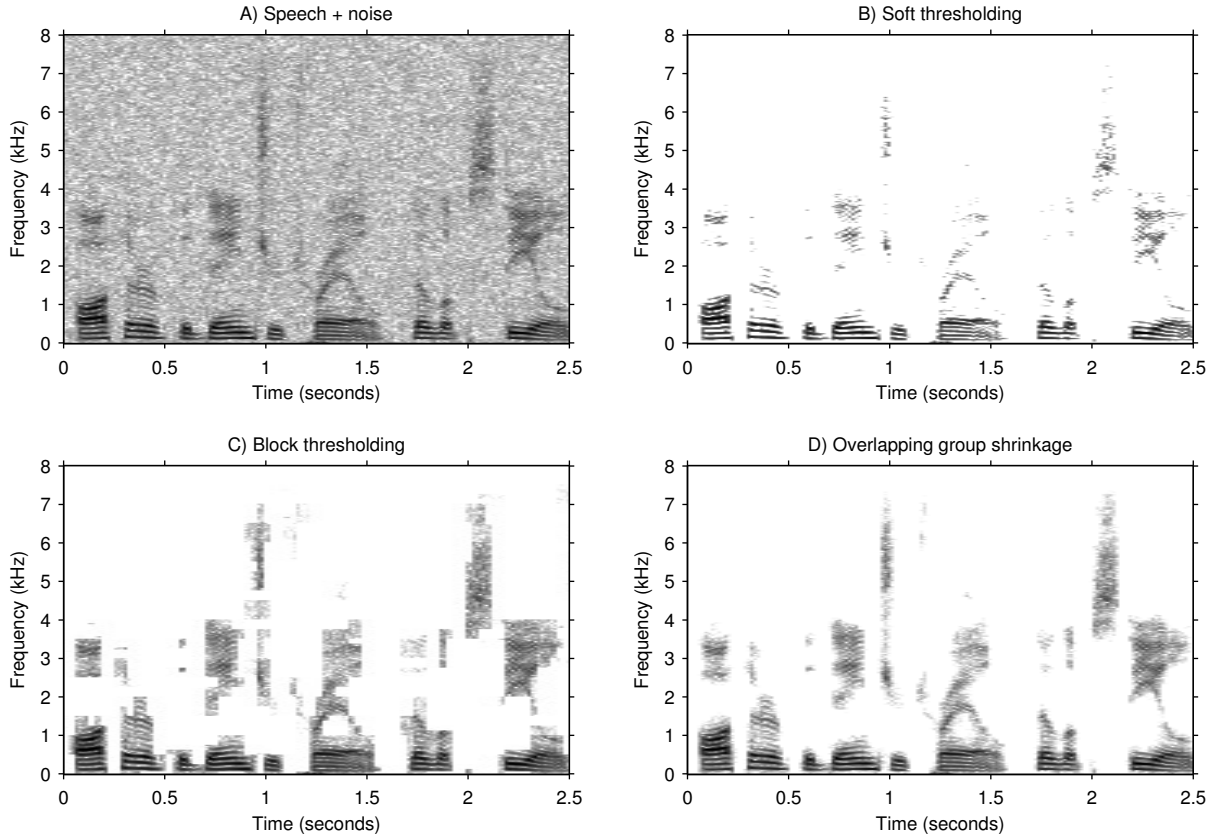

Figure 6: Spectrograms of (a) noisy speech and (b) result of soft thresholding. Spectrograms denoised by (c) block thresholding [73] and (d) overlapping group shrinkage (OGS). Gray scale represents decibels.

but it uses a model based on fully overlapping blocks. Some other recent algorithms exploiting structured time-frequency sparsity of speech/audio are based on Markov models [21, 26, 72].

To apply OGS to speech denoising, we solve (2), where $\mathbf{y} = \Phi\mathbf{s}$ is the complex short-time Fourier transform (STFT) of the noisy speech $\mathbf{s}$, Φ is the STFT, $\mathbf{x}$ is the STFT coefficients to be determined, and $R(\mathbf{x})$ is (4). The estimated speech is then given by $\hat{\mathbf{x}} = \Phi^H\mathbf{x}$. To minimize (2), we use the two-dimensional form of the OGS algorithm applied to the STFT coefficients.

For illustration, consider the noisy speech illustrated in Fig. 6a. The noisy speech is a male utterance sampled at 16 kHz, corrupted by additive white Gaussian noise with SNR 10 dB.[4] The STFT is calculated with 50% overlapping blocks of length of 512 samples. Figure 6b illustrates the STFT obtained by soft thresholding the noisy STFT, with threshold T selected so as to reduce the noise standard deviation down to 0.1% of its value. From Table 3, we obtain $T = 3.26\sigma$, where σ is the noise standard deviation in the STFT domain. The noise is sufficiently suppressed so that musical noise is not audible; however, the signal is distorted due to the relatively high threshold used. The spectrogram in Fig. 6b is overly thinned.

Figure 6c illustrates the result of block thresholding [73] using the software provided by the authors.[5] The SNR is 15.35 dB. It can be seen that block thresholding (BT) produces blocking artifacts in the spectrogram. Some auditory artifacts are perceptible in the BT denoised speech.

Figure 6d illustrates the result of overlapping group shrinkage (OGS) applied to the noisy STFT. We

[4]Speech file `arctic_a0001.wav` downloaded from `http://www.speech.cs.cmu.edu/cmu_arctic/cmu_us_bdl_arctic/wav`.
[5]`http://www.cmap.polytechnique.fr/~yu/research/ABT/samples.html`

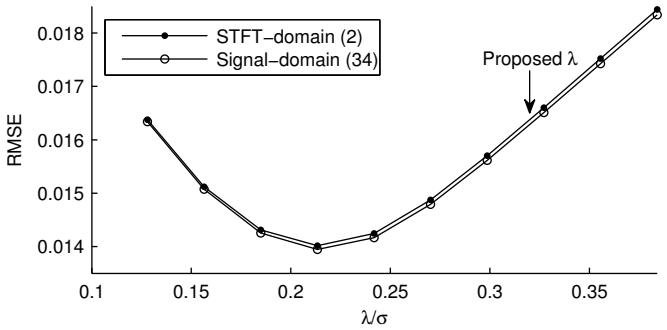

Figure 7: Comparison of RMSE using STFT-domain cost function (2) and signal-domain cost function (35). Formulation (35) gives a slightly better RMSE. The proposed value ($\lambda = 0.32\sigma$) is indicated by the arrow.

used 25 iterations of the OGS algorithm. Based on listening to speech signals denoised with various group sizes, we selected a group size 8×2 (i.e., eight frequency bins $\times$ two time bins). Other group sizes may be more appropriate for other sampling rates and STFT block lengths. As in the soft thresholding experiment, the parameter λ was selected so as to reduce the noise standard deviation down to 0.1% of its value. From Table 3, we obtain $\lambda = 0.32\sigma$. The SNR of the denoised speech is 13.77 dB. While the SNR is lower than block thresholding, the artifacts of the OGS-denoised speech are less audible and musical noise is not audible.

It was found in [73] that empirical Wiener post-processing (EWP), introduced in [32], improves the result of the block thresholding (BT) algorithm. This post-processing, which is computationally very simple, improves the result of OGS by an even greater degree than for BT, as measured by SNR improvement. The Wiener post-processing raises the SNR for BT from 15.35 dB to 15.75 dB, while it raises the SNR for OGS from 13.77 dB to 15.63 dB. Hence, the two methods give almost the same SNR after Wiener post-processing. The substantial SNR improvement in the case of OGS can be explained as follows: the OGS algorithm has the effect of slightly shrinking (attenuating) large coefficients which produces a bias and negatively affects the SNR of the denoised signal. The Wiener post-processing procedure largely corrects that bias. It has the effect of rescaling (slightly amplifying) the large coefficients appropriately.

Signal-domain data fidelity. Note that, due to the STFT not being an orthonormal transform, the noise in the STFT domain is not white, even when it is white in the signal domain. Therefore, instead of problem (2), it is reasonable to solve

$$\min_{\mathbf{x}} \; \frac{1}{2} \|\mathbf{s} - \Phi^H \mathbf{x}\|_2^2 + \lambda \, R(\mathbf{x}), \tag{35}$$

which is consistent with the white noise assumption (the data fidelity term is in the signal domain). Problem (35) can be solved by proximal methods (e.g. forward-backward splitting (FBS)) [4,15,16] or ADMM methods [7, 20, 27].

We note that the solving (35) takes more computation than solving (2). Formulation (2) is, by definition, a single proximal operator (which OGS implements). On the other hand, solving (35) (using ADMM or FBS, etc.) requires the repeated application of the proximal operator, interlaced with Φ and Φ^H (forward and inverse STFTs). In our experiment, solving (35) using ADMM took 5.95 seconds, while solving (2) took 0.25 seconds. We ran 25 iterations of OGS in each ADMM iteration to implement the proximal operator.

Figure 7 shows the RMSE as a function of λ for each of (2) and (35). Note that (35) attains a better RMSE (higher SNR) than (2), but the difference is marginal and imperceptible. We also note that, for both formulations (2) and (35), when λ is optimized so as to obtain the minimum RMSE (best SNR), 'musical

Table 5: Output SNR of several speech enhancement algorithms, without (w/o) and with empirical Wiener post-processing (EWP).

	SNR (dB)	
Method	w/o EWP	EWP
spectral subtract. (SS) [5]	13.50	14.64
log MMSE alg. (LMA) [14]	13.10	14.93
subspace alg. (SUB) [36]	13.68	15.77
block-thresh. (BT) [73]	15.35	15.75
persistent shrink. (PS) [69]	13.63	15.77
OGS (proposed)	13.77	15.63

noise' is clearly audible in the denoised speech. In terms of perceptual quality, a higher value of λ gives a much better result, even though the RMSE is worse. Our proposed method for setting λ, indicated by the arrow in Fig. 7, gives a higher RMSE, but the result is perceptually preferable due to the suppression of 'musical noise'. Note that the increase in RMSE, due to using a higher λ so as to avoid musical noise, outweighs the increase in RMSE due to using (2) instead of (35). Due to the imperceptible difference between (2) and (35), and the higher computational complexity of the latter, the formulation (2) appears suitable for speech denoising using OGS.

Further comparisons. This speech denoising example is intended as an illustrative example of OGS rather than state-of-the-art speech denoising. However, to provide further context, the output SNR of several algorithms, with and without empirical Wiener post-processing (EWP), are summarized in Table 5. The additional algorithms are: spectral subtraction (SS) [5], the log MMSE algorithm (LMA) [14], the subspace algorithm (SUB) [36], and persistent shrinkage (PS) [69]. For SS, LMA, and SUB, we used the MATLAB software provided in Ref. [44]. For PS, we used the software provided by the authors.[6] Without EWP, BT achieves the highest SNR of 15.35 dB. However, the BT has slightly audible artifacts as noted above, as does PS. The artifacts of SUB and OGS are less audible. The artifacts of SS and LMA are clearly audible.

EWP improves the SNR of SUB and LMA by 1.14 dB and 1.83 dB; however, perceptual artifacts are still clearly audible. EWP improves BT by 0.4 dB, and has a minor impact on perceptual quality. EWP improves SUB, PS and OGS by 2.09 dB, 2.14 dB, and 1.86 dB, and slightly reduces the audible artifacts. We note that the form of EWP used in PS is a *persistent* form introduced in [69]. With EWP, four methods (BT, SUB, PS, and OGS) achieve almost the same SNR (with varying degrees of audible musical noise). However, SUB has a high computational complexity due to eigenvalue factorization.

5 Conclusion

This paper introduces a computationally efficient algorithm for denoising signals with group sparsity structure.[7] In this approach, 'overlapping group shrinkage' or OGS, the groups are fully overlapping and the algorithm is translation-invariant. The method is based on the minimization of a convex function.

A procedure is described for the selection of the regularization parameter λ. The procedure is based on attenuating the noise to a specified level without regard to the statistical properties of the signal of interest. In this sense, the procedure for setting λ is not Bayesian; it does not depend on the signal to be estimated.

[6] http://homepage.univie.ac.at/monika.doerfler/StrucAudio.html

[7] MATLAB software to implement the OGS algorithm and to reproduce the figures in the paper are online at http://eeweb.poly.edu/iselesni/ogs/.

Even though the described procedure for setting λ is conceptually simple, it does not admit the use of explicit formulas for λ because, in part, the OGS function does not itself have an explicit formula (it being defined as the solution to a minimization problem). The procedure to set λ is based on analyzing the output of OGS when it is applied to an i.i.d. standard normal signal, and can be implemented in practice by off-line computation of tables such as Tables 2 and 3. Adopting recent approaches for regularization parameter selection [19] provides another potential approach to be investigated.

The paper illustrates the use of overlapping group shrinkage for speech denoising. The OGS algorithm is applied to the STFT of the noisy signal. Compared to the block thresholding algorithm [73], the OGS algorithm gives similar SNR when Wiener post-processing is used. However, the OGS denoised speech has fewer perceptual artifacts.

Current work includes the extension of OGS to non-convex penalty functions that promote sparsity more strongly than the convex penalty function in (4). In particular, as developed in [63], it is possible to prescribe non-convex penalty functions such that the total cost function F in (2) is still convex. With this approach, large amplitude coefficients are less attenuated resulting in less distortion. While, due to its convexity, the cost function, F, can be minimized via convex optimization and the problem of local minima is avoided, unlike methods based on non-convex optimization.

A Convergence and Zero-locking

Figure 8 illustrates the non-global convergence behavior of the OGS algorithm. For clarity, to produce Fig. 8, we have used a signal of length only 2 samples with group size 2. Specifically, $\mathbf{y} = [3, 4] \in \mathbb{R}^2$, $\lambda = 1$, $K = 2$, $N = 2$.

The OGS cost function, $F : \mathbb{R}^N \to \mathbb{R}$, is strictly convex, so the minimizer is unique. Yet, as shown in Fig. 8, the OGS algorithm has four fixed points. The fixed point $(1, 0)$ has the attraction set $\{(x_0, 0) : x_0 > 0\}$. The fixed point $(0, 2)$ has the attraction set $\{(0, x_1) : x_1 > 0\}$. The fixed point $(0, 0)$ has the attraction set $\{(0, 0)\}$, i.e., it is an isolated fixed point. The fixed point $(1.45, 2.17)$ has the attraction set $\{(x_0, x_1) : x_0 > 0, x_1 > 0\}$.

Although the OGS algorithm has four fixed points, only one of them is the minimizer of F. The value of F at the fixed points are: $F(1, 0) = 12.0$. $F(0, 2) = 10.5$, $F(0, 0) = 12.5$, $F(1.45, 2.17) = 9.1$. The fixed point $(1.45, 2.17)$ is the minimizer of F.

The three extraneous fixed points are consequences of the zero-locking phenomenon; they are not local minimizers of F (c.f. F is strictly convex). Figure 8 shows that if some or all of the components of $\mathbf{x}$ are initialized to zero, then the OGS algorithm may fail to converge to the minimizer of F. In general, an N-point signal $\mathbf{x} \in \mathbb{R}^N$ may have up to 2^N fixed points. However, if some components of the minimizer $\mathbf{x}^*$ are zero (when $\mathbf{x}^*$ is sparse, many are zero), then there will be fewer fixed points.

The extraneous fixed points are avoided in practice by suitable non-zero initialization, as discussed in Sec. 2.3. Therefore, the presence of extraneous fixed points does not impede the convergence of the OGS algorithm in practice.

Figure 8 may suggest that the extraneous fixed points are easily identified as extraneous due to their having zero-valued components. However, in general, the minimizer, $\mathbf{x}^*$, may have many zero components and it is not simple to identify a fixed point as extraneous (except by running the OGS algorithm with a non-zero initialization and observing the solution to which it converges). That is to say, identifying a fixed point of OGS as extraneous is no easier than solving the original OGS problem.

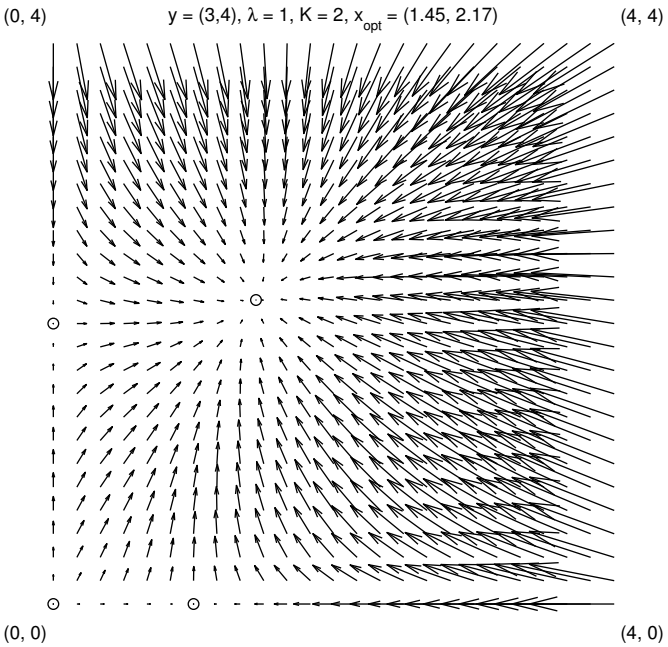

Figure 8: Illustration of convergence behavior for OGS for a two point signal. There are 4 fixed points, only one of which minimizes the OGS cost function.

References

[1] A. Achim and E. E. Kuruoğlu. Image denoising using bivariate α-stable distributions in the complex wavelet domain. *IEEE Signal Processing Letters*, 12(1):17–20, January 2005.

[2] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Structured sparsity through convex optimization. Technical report, Hal-00621245, 2011.

[3] I. Bayram. Mixed norms with overlapping groups as signal priors. In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, pages 4036–4039, May 2011.

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

[5] M. Berouti, R. Schwartz, and J. Makhoul. Enhancement of speech corrupted by acoustic noise. In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, volume 4, pages 208–211, April 1979.

[6] S. Boll. Suppression of acoustic noise in speech using spectral subtraction. *IEEE Trans. on Acoust., Speech, Signal Proc.*, 27(2):113–120, April 1979.

[7] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[8] T. Cai and B. W. Silverman. Incorporating information on neighboring coefficients into wavelet estimation. *Sankhya: Ind. J. Stat. B*, 63:127–148, 2001.

[9] O. Cappe. Elimination of the musical noise phenomenon with the Ephraim and Malah noise suppressor. *IEEE Trans. Speech Audio Process.*, 2(2):345–349, April 1994.

[10] L. Chaari, J.-C. Pesquet, J.-Y. Tourneret, P. Ciuciu, and A. Benazza-Benyahia. A hierarchical Bayesian model for frame representation. *IEEE Trans. Signal Process.*, 58(11):5560–5571, November 2010.

[11] S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM J. Sci. Comput.*, 20(1):33–61, 1998.

[12] X. Chen, Q. Lin, S. Kim, J. G. Carbonell, and E. P. Xing. Smoothing proximal gradient method for general structured sparse regression. *Ann. Appl. Stat.*, 6(2):719–752, 2012.

[13] D. Cho and T. D. Bui. Multivariate statistical modeling for image denoising using wavelet transforms. *Signal Processing: Image Communications*, 20(1):77–89, January 2005.

[14] I. Cohen. Optimal speech enhancement under signal presence uncertainty using log-spectral amplitude estimator. *IEEE Signal Processing Letters*, 9(4):113–116, April 2002.

[15] 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.

[16] P. L. Combettes and V. R. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling & Simulation*, 4(4):1168–1200, 2005.

[17] S. F. Cotter, B. D. Rao, K. Engan, and K. Kreutz-Delgado. Sparse solutions to linear inverse problems with multiple measurement vectors. *IEEE Trans. Signal Process.*, 53(7):2477–2488, July 2005.

[18] 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.

[19] C. Deledall, S. Vaiter, G. Peyre, J. Fadili, and C. Dossal. Proximal splitting derivatives for risk estimation. Technical report, Hal-00670213, 2012.

[20] W. Deng, W. Yin, and Y. Zhang. Group sparse optimization by alternating direction method. *Rice University CAAM Technical Report TR11-06, 2011*, 2011.

[21] O. Dikmen and A. T. Cemgil. Gamma Markov random fields for audio source modeling. *IEEE Trans. on Audio, Speech, and Lang. Proc.*, 18(3):589–601, March 2010.

[22] D. L. Donoho. De-noising by soft-thresholding. *IEEE Trans. on Information Theory*, 41(3):613–627, May 1995.

[23] M. Elad, M. A. T. Figueiredo, and Y. Ma. On the role of sparse and redundant representations in image processing. *Proc. IEEE*, 98(6):972–982, June 2010.

[24] Y. C. Eldar and M. Mishali. Robust recovery of signals from a structured union of subspaces. *IEEE Trans. Inform. Theory*, 55(11):5302–5316, 2009.

[25] J. M. Fadili and L. Boubchir. Analytical form for a Bayesian wavelet estimator of images using the Bessel K form densities. *IEEE Trans. Image Process.*, 14(2):231–240, February 2005.

[26] C. Fevotte, B. Torresani, L. Daudet, and S. J. Godsill. Sparse linear regression with structured priors and application to denoising of musical audio. *IEEE Trans. on Audio, Speech, and Lang. Proc.*, 16(1):174–185, January 2008.

[27] M. Figueiredo and J. Bioucas-Dias. An alternating direction algorithm for (overlapping) group regularization. In *Signal Processing with Adaptive Sparse Structured Representations (SPARS)*, 2011.

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

[29] M. Figueiredo and R. Nowak. Wavelet-based image estimation: An empirical Bayes approach using Jeffrey's noninformative prior. *IEEE Trans. Image Process.*, 10(9):1322–1331, September 2001.

[30] H. Gao. Wavelet shrinkage denoising using the nonnegative garrote. *J. Comput. Graph. Statist.*, 7:469–488, 1998.

[31] P. V. Gehler and M. Welling. Product of "edge-perts". *Advances in Neural Information Processing System 18*, pages 419–426, 2005.

[32] S. Ghael, A. M. Sayeed, and R. G. Baraniuk. Improved wavelet denoising via empirical Wiener filtering. In *SPIE Tech. Conf. Wavelet Appl. Signal Proc.*, San Diego, July 1997.

[33] B. Goossens, A. Pizurica, and W. Philips. Image denoising using mixtures of projected Gaussian scale mixtures. *IEEE Trans. Image Process.*, 18(8):1689 –1702, August 2009.

[34] I. F. Gorodnitsky and B. D. Rao. Sparse signal reconstruction from limited data using FOCUSS: a re-weighted minimum norm algorithm. *IEEE Trans. Signal Process.*, 45(3):600–616, March 1997.

[35] R. C. Hendriks and R. Martin. MAP estimators for speech enhancement under normal and Rayleigh inverse Gaussian distributions. *IEEE Trans. on Audio, Speech, and Lang. Proc.*, 15(3):918 –927, March 2007.

[36] Y. Hu and P. C. Loizou. A generalized subspace approach for enhancing speech corrupted by colored noise. *IEEE Trans. on Acoust., Speech, Signal Proc.*, 11(4):334–341, July 2003.

[37] A. Hyvärinen. Sparse code shrinkage: Denoising of non-Gaussian data by maximum likelihood estimation. *Neural Computation*, 11:1739–1768, 1999.

[38] L. Jacob, G. Obozinski, and J.-P. Vert. Group lasso with overlap and graph lasso. *Proc. 26th Annual Int. Conf. Machine Learning*, 2009.

[39] R. Jenatton, J.-Y. Audibert, and F. Bach. Structured variable selection with sparsity-inducing norms. *J. Mach. Learning Research*, 12:2777–2824, October 2011.

[40] R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for sparse hierarchical dictionary learning. In *Proc. Int. Conf. on Machine Learning (ICML)*, 2010.

[41] M. Kowalski. Sparse regression using mixed norms. *J. of Appl. and Comp. Harm. Analysis*, 27(3):303–324, 2009.

[42] M. Kowalski and B. Torrésani. Sparsity and persistence: mixed norms provide simple signal models with dependent coefficients. *Signal, Image and Video Processing*, 3(3):251–264, 2009.

[43] J. Liu and P. Moulin. Information-theoretic analysis of interscale and intrascale dependencies between image wavelet coefficients. *IEEE Trans. Image Process.*, 10(11):1647–1658, November 2001.

[44] P. C. Loizou. *Speech enhancement: theory and practice*. CRC Press, 2007.

[45] D. G. Luenberger and Y. Ye. *Linear and nonlinear programming*. Springer, 3rd edition, 2008.

[46] S. Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Trans. Pattern Anal. and Machine Intel.*, 11(7):674–693, July 1989.

[47] R. Martin. Speech enhancement based on minimum mean-square error estimation and supergaussian priors. *IEEE Trans. Speech Audio Process.*, 13(5):845–856, September 2005.

[48] R. McAulay and M. Malpass. Speech enhancement using a soft-decision noise suppression filter. *IEEE Trans. on Acoust., Speech, Signal Proc.*, 28(2):137–145, April 1980.

[49] M. K. Mihcak, I. Kozintsev, K. Ramchandran, and P. Moulin. Low-complexity image denoising based on statistical modeling of wavelet coefficients. *IEEE Signal Processing Letters*, 6(12):300–303, December 1999.

[50] D. E. Molina, D. Gleich, and M. Datcu. Gibbs random field models for model-based despeckling of SAR images. *IEEE Geoscience and Remote Sensing Letters*, 7(1):73–77, January 2010.

[51] S. Mosci, S. Villa, A. Verri, and L. Rosasco. A primal-dual algorithm for group sparse regularization with overlapping groups. In *Advances in Neural Information Processing Systems 23*, pages 2604–2612, 2010.

[52] G. Obozinski, L. Jacob, and J. P. Vert. Group lasso with overlaps: the latent group lasso approach. Technical report, HAL-Inria-00628498, 2011.

[53] J. Oliveira, J. Bioucas-Dias, and M. A. T. Figueiredo. Adaptive total variation image deblurring: A majorization-minimization approach. *Signal Processing*, 89(9):1683–1693, September 2009.

[54] G. Peyre and J. Fadili. Group sparsity with overlapping partition functions. In *Proc. European Sig. Image Proc. Conf. (EUSIPCO)*, Aug. 29 - Sept. 2 2011.

[55] A. Pizurica, W. Philips, I. Lemahieu, and M. Acheroy. A joint inter- and intrascale statistical model for Bayesian wavelet based image denoising. *IEEE Trans. Image Process.*, 11(5):545–557, May 2002.

[56] J. Portilla, V. Strela, M. J. Wainwright, and E. P. Simoncelli. Image denoising using scale mixtures of Gaussians in the wavelet domain. *IEEE Trans. Image Process.*, 12(11):1338–1351, November 2003.

[57] N. Pustelnik, C. Chaux, and J. Pesquet. Parallel proximal algorithm for image restoration using hybrid regularization. *IEEE Trans. Image Process.*, 20(9):2450–2462, September 2011.

[58] H. Rabbani, R. Nezafat, and S. Gazor. Wavelet-domain medical image denoising using bivariate Laplacian mixture model. *Trans. on Biomed. Eng.*, 56(12):2826–2837, December 2009.

[59] B. D. Rao, K. Engan, S. F. Cotter, J. Palmer, and K. Kreutz-Delgado. Subset selection in noise based on diversity measure minimization. *IEEE Trans. Signal Process.*, 51(3):760–770, March 2003.

[60] B. D. Rao and K. Kreutz-Delgado. An affine scaling methodology for best basis selection. *IEEE Trans. Signal Process.*, 47(1):187–200, January 1999.

[61] N. S. Rao, R. D. Nowak, S. J. Wright, and N. G. Kingsbury. Convex approaches to model wavelet sparsity patterns. In *Proc. IEEE Int. Conf. Image Processing*, pages 1917–1920, September 2011.

[62] I. W. Selesnick. The estimation of Laplace random vectors in additive white Gaussian noise. *IEEE Trans. Signal Process.*, 56(8):3482–3496, August 2008.

[63] I. W. Selesnick and I. Bayram. Sparse signal estimation by maximally sparse convex optimization. http://arxiv.org/abs/1302.5729, February 2013.

[64] L. Sendur and I. W. Selesnick. Bivariate shrinkage functions for wavelet-based denoising exploiting interscale dependency. *IEEE Trans. Signal Process.*, 50(11):2744–2756, November 2002.

[65] L. Sendur and I. W. Selesnick. Multivariate shrinkage functions for wavelet-based denoising. In *Asilomar conf. signals, systems and computers*, volume 1, pages 953–957, November 2002.

[66] J. M. Shapiro. Embedded image coding using zerotrees of wavelet coefficients. *IEEE Trans. on Acoust., Speech, Signal Proc.*, 41(12):3445–3462, December 1993.

[67] P.-L. Shui and Y.-B. Zhao. Image denoising algorithm using doubly local Wiener filtering with block-adaptive windows in wavelet domain. *Signal Processing*, 87(7):1721–1734, July 2007.

[68] K. Siedenburg and M Dörfler. Structured sparsity for audio signals. In *Proc. 14th Int. Conf. Digital Audio Effects (DAFx-11)*, September 2011.

[69] K. Siedenburg and M. Dörfler. Persistent time-frequency shrinkage for audio denoising. *J. Audio Eng. Soc.*, 61(1):29–38, 2013.

[70] E. Simoncelli and B. Olshausen. Natural image statistics and neural representation. *Annual Review of Neuroscience*, 24:1193–1216, May 2001.

[71] R. Tibshirani. Regression shrinkage and selection via the lasso. *J. Roy. Statist. Soc., Ser. B*, 58(1):267–288, 1996.

[72] P. J. Wolfe, S. J. Godsill, and W.-J. Ng. Bayesian variable selection and regularisation for time-frequency surface estimation. *J. R. Statist. Soc., Series B*, 66(3):575–589, 2004.

[73] G. Yu, S. Mallat, and E. Bacry. Audio denoising by time-frequency block thresholding. *IEEE Trans. Signal Process.*, 56(5):1830–1839, May 2008.

[74] L. Yuan, J. Liu, and J. Ye. Efficient methods for overlapping group lasso. In *Advances in Neural Information Processing Systems 24*, pages 352–360. 2011.

[75] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68(1):49–67, February 2006.

[76] R. Zdunek and A. Cichocki. Improved M-FOCUSS algorithm with overlapping blocks for locally smooth sparse signals. *IEEE Trans. Signal Process.*, 56(10):4752–4761, October 2008.

Supplementary Material

Sound files from the speech denoising example:

`speech_noisy.wav`	speech plus noise
`speech_soft.wav`	soft thresholding
`speech_ogshrink.wav`	overlapping group shrinkage (no Wiener post-processing)
`speech_ogshrink_wp.wav`	overlapping group shrinkage (with Wiener post-processing)
`speech_blockthresh.wav`	block thresholding (no Wiener post-processing)
`speech_blockthresh_wp.wav`	block thresholding (with Wiener post-processing)

MATLAB programs for the implementation of overlapping group shrinkage (1D).

```
function [a, cost] = ogshrink(y, K, lam, Nit)
% [a, cost] = ogshrink(y, K, lam, Nit);
% Overlapping group shrinkage (OGS)
% Minimizes the cost function with respect to a
%
% cost = 0.5 * sum(abs(y - a).^2) + lam * sum(sqrt(conv(abs(a).^2, ones(1,K))));
%
% INPUT
%   y   : 1-D  noisy signal (vector)
%   K   : size of group
%   lam : regularization parameter
%   Nit : number of iterations
%
% OUTPUT
%   a    : output (denoised signal)
%   cost : cost function history

% Po-Yu Chen and Ivan Selesnick
% Polytechnic Institute of New York University
% New York, USA
% March 2012

a = y;                  % initialize
h = ones(1,K);          % for convolution
cost = zeros(1,Nit);
i = (a ~= 0);
for it = 1:Nit
    r = sqrt(conv(abs(a).^2, h));
    cost(it) = 0.5*sum(abs(y - a).^2) + lam * sum(r);
    v = 1 + lam*conv(1./r, h);
    v = v(K:end+1-K);
    % In newer MATLAB versions, the above 2 lines can be replaced with 1 line:
    % v = 1 + lam*conv(1./r, h, 'valid');
    a = y./v;
end
```

MATLAB programs for the implementation of overlapping group shrinkage (2D).

```
function [a, cost] = ogshrink2(y, K1, K2, lam, Nit)
% [a, cost] = ogshrink2(y, K1, K2, lam, Nit);
% 2D overlapping group shrinkage (OGS)
% Minimizes the cost function with respect to a
%
% cost = 0.5*sum(sum(abs(y - a).^2)) + lam * sum(sqrt(conv(abs(a).^2, ones(K1,K2))));
%
% INPUT
%   y      : 2-D  noisy signal (2D array)
%   K1, K2 : size of group
%   lam    : regularization parameter
%   Nit    : number of iterations
%
% OUTPUT
%   a    : output (denoised signal)
%   cost : cost function history

% Po-Yu Chen and Ivan Selesnick
% Polytechnic Institute of New York University
% New York, USA
% March 2012

a = y;                  % initialize
h1 = ones(K1,1);        % for convolution
h2 = ones(K2,1);        % for convolution
cost = zeros(1,Nit);
for it = 1:Nit
    r = sqrt(conv2(h1, h2, abs(a).^2));
    cost(it) = 0.5*sum(sum(abs(y - a).^2)) + lam * sum(r(:));
    v = 1 + lam*conv2(h1, h2, 1./r);
    v = v(K1:end+1-K1, K2:end+1-K2);
    % In newer MATLAB versions, the above 2 lines can be replaced with 1 line:
    % v = 1 + lam*conv2(h1, h2, 1./r, 'valid');
    a = y./v;
end
```

```
function a = soft(y, T)
% Soft-threshold function (for real or complex data)
% a = soft(y, T)
%
% INPUT
%    y : data
%    T : threshold
%
% If y and T are both multidimensional, then they must be of the same size.

a = max(1 - T./abs(y), 0) .* y;
```