Piecewise-polynomial Curve Fitting Using Group Sparsity

Michaela Novosadová and Pavel Rajmic
Signal Processing Laboratory (SPLab)
Dept. of Telecommunications, Faculty of Electrical Engineering and Communication
Brno University of Technology, Technická 12, 616 00 Brno, Czech Republic
Email: xnovos11@stud.feec.vutbr.cz, rajmic@feec.vutbr.cz

***Abstract*—We present a method for segmenting onedimensional signal, whose segments are modeled as polynomials, and which is corrupted by an additive noise. The method is based on sparse modeling and its formulation as a convex optimization problem is solved by proximal algorithms. We perform experiments on simulated data, discuss the results and suggest future directions that could lead to even better results.**

I. Introduction

Segmentation of a signal is one of the most important applications in digital signal processing. In this article, we use convex optimization to segment onedimensional signal, i.e. to automatically find the partitioning of such a signal into its consistent segments. In our case, the clean, underlying signal is assumed to break into several polynomial segments, nonoverlapping and independent of each other, whose number is assumed to be considerably lower than the total number of the observed signal samples. This last fact suggests utilizing sparse signal processing techniques.

A number of real-world time series can be considered piecewise polynomial, and thus, this method is of practical interest per se. Nevertheless, this paper can be seen as an intermediate step towards segmentation of images. Indeed, the methods which will be presented can be smoothly generalized to a higher dimension, and the piecewise-polynomial nature of segments is still valid, since [1], [2] showed that images, to a great extent, can be modeled as piecewise smooth 2D-functions.

The recent publication [3], coping with segmentation of images, was actually the motivation for this work. The authors of [3] use greedy [4] approach enabling the state-of-the-art results, and our goal was to explore how the convex relaxations of sparsity measures [5] will compare to it.

Finding consistent segments in a signal/time series is an old problem, referred to as the change point problem in the field of statistics. Most resources cope with finding change points in noisy, piecewise *constant* signal, which is either singlevariate [6] or multivariate [7], [8] (which is the case where most of the signals or all of them change simultaneously). In a more general context of detecting sudden changes in the underlying, latent *parameters* we mention works [9] and [10]. Article [11] utilizes sparse techniques, however, it considers polynomial model for the *entire* signal, with additive sparse ramp discontinuities.

When the signal is assumed to lack jumps (i.e. the segments borderpoints have to coincide), efficient methods have been presented, based on sparse modeling, see [12] or [13] in a more general setup.

Although our segmentation problem relates to the mentioned change-point problems, it is different, as will be revealed shortly in detail. To the best of our knowledge, there is no other work apart from [3] which would use *overparametrization* for segmenting the signal, and apart from [14] which, however, constructs nonconvex optimization problems.

We have already made some work in the proposed direction. Our publication [15] discussed the use of convex total variation (TV) regularizers for the segmentation task. This work extends this previous paper in three directions: First, in [15] we only worked with piecewise linear signals, while here we allow polynomials of general degree. Second, sparse regularization is shifted to the joint-sparse, ensuring better breakpoint detection. And third, we present a more sophisticated method of breakpoints assessment.

II. Problem formulation

We work with one-dimensional piecewise polynomial signal, denoted $\mathbf{y} = [y_1, \dots, y_N]^\top \in \mathbb{R}^N$. Consecutive segments do not have to meet at joint knots—jumps in $\mathbf{y}$ are allowed. Within segment $s \in \{1, \dots, S\}$, entries of $\mathbf{y}$ can be described by $k+1$ parametrization coefficients $x_{s,0}, \dots, x_{s,k}$ such that the i-th element of $\mathbf{y}$ is

$$y_i = x_{s,0} + x_{s,1} \cdot i + x_{s,2} \cdot i^2 + \cdots + x_{s,k} \cdot i^k + \varepsilon_i, \quad (1)$$

where ε_i stands for perturbations by uncorrelated Gaussian noise with zero mean and positive variance. It is clear that $x_{s,0}$ represents local intercept, $x_{s,1}$ represents local slope etc. In vector notation, we can model the signal as

$$\mathbf{y} = [\mathbf{I}\ \mathbf{D}^1 \dots \mathbf{D}^k] \begin{bmatrix} \mathbf{x}_0 \\ \vdots \\ \mathbf{x}_k \end{bmatrix} + \mathbf{e} \quad (2)$$

where $\mathbf{x}_0, \dots, \mathbf{x}_k$ are (column) parametrization vectors in $\mathbb{R}^N$, $\mathbf{I} = \mathbf{I}_N$ is the identity matrix, $\mathbf{D} = \mathrm{diag}(1, 2, \dots, N)$ is a diagonal matrix with linearly growing entries, $\mathbf{D}^j$ is its power (and therefore $\mathbf{D}^0 = \mathbf{I}$), and $\mathbf{e}$ represents the noise vector. At places, we will use a shortened notation $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{e}$ instead of (2) for simplicity.

978-1-4673-8818-4/16/$31.00 ©2016 IEEE

ISSN: 2157-023X

Thanks to the above mentioned formulation, it is obvious that signal generation by $\mathbf{A} \cdot \mathbf{x}$ is heavily overparametrized. On the other hand, consider that if the signal consists of S consecutive segments built from polynomials up to degree k, the parametrization vectors $\mathbf{x}_i$ stay piecewise constant within each segment.

This fact, together with the assumption $S \ll N$, motivates the use of a measure of sparsity for the denoising and curve fitting problem. Piecewise constant vectors $\mathbf{x}_i$ suggest that these vectors are sparse under the difference operation, ∇. Indeed, in our paper [15] we used the total variation (TV) penalty in order to recover piecewise constant parametrization vectors. TV of a vector $\mathbf{z}$ is defined by $\mathrm{TV}(\mathbf{z}) = \|\nabla \mathbf{z}\|_1 = \sum_{j=1}^{N-1} |z_{j+1} - z_j|$, where the ℓ_1-norm enforces $\mathbf{z}$ to be approximately sparse in differences. The approach from [15] utilizes TV to treat each $\mathbf{x}_i$ separately, however, according to the model, all parametrization vectors potentially jump in their value only at the segment breakpoints. We therefore assume that the changepoints (jumps) in all $\mathbf{x}_i$s lie at the same positions in this paper.

To this end, we formulate the recovery problem using the so-called mixed norm [16] as follows:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \ \frac{1}{2}\|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \|[\tau_0 \nabla \mathbf{x}_0, \ldots, \tau_k \nabla \mathbf{x}_k]\|_{21} \quad (3)$$

where $k+1$ regularization weights $\tau_0, \ldots, \tau_k$ are used, corresponding to individual polynomial degrees; these scalars should be set carefully according to the noise level and prior experience. We use the ℓ_{21}-norm, whose input is a matrix. The ℓ_{21}-norm, used as a regularizer, does promote sparsity across matrix' columns, and does not do it across the rows. More precisely, $\|[\cdot, \ldots, \cdot]\|_{21}$ computes the ℓ_1-norm of vector consisting of ℓ_2-norms of the matrix' rows. In our case, columns of the matrix are the difference vectors $\tau_j \nabla \mathbf{x}_j$, and therefore the *joint* sparsity of differences is enforced. This means that differences should be zero at most matrix' positions, and nonzero values should be concentrated in a few rows, i.e. in points which correspond to breaks in the model.

Vector $\hat{\mathbf{x}}$ contains the achieved optimizers, each $\hat{\mathbf{x}}_i \in \mathbb{R}^N$, and (possibly joint) nonzeros values in $\nabla \hat{\mathbf{x}}_i$s indicate the possible segment borders (changepoints).

III. Algorithms

A. Proximal splitting methods

Proximal splitting methodology is a tool for iterative solution of convex minimization problems [17]. For the purpose of solving (3), we will need an algorithm able to

$$\text{minimize} \ \ f_1(\mathbf{x}) + f_2(L\mathbf{x}), \quad (4)$$

where both f_1 and f_2 are convex functions, f_1 can be smooth and f_2 can be nonsmooth, and L is any linear operator. Based on the properties of the functions, proximal algorithms perform iterations involving evaluation of their individual gradients and/or proximal operators, which is much simpler than minimization of the composite functional by other means. Such first-order methods are in particular suitable for large-scale problems [18]. It is proven that proximal splitting algorithms provide convergence to the optimal value. The speed of convergence is influenced by character of functions f_1 and f_2 and by the parameters used in the algorithms.

There is a number of choices for solving (4). We used the Forward-backward based primal-dual algorithm (FBBPD) [19] and the Chambolle-Pock (ChP) [18] algorithm, see the description later.

B. Proximal operators

A proximal operator of a function h maps $\mathbf{x} \in \mathbb{R}^N$ to another vector in $\mathbb{R}^N$,

$$\underset{h}{\mathrm{prox}}(\mathbf{x}) = \arg\min_{\mathbf{z} \in \mathbb{R}^N} \ h(\mathbf{z}) + \frac{1}{2}\|\mathbf{z} - \mathbf{x}\|_2^2. \quad (5)$$

We will make use of proximal operators of two specific functions in this paper. First, we have [17], [20] that

$$\underset{\frac{\tau}{2}\|\mathbf{A}\cdot - \mathbf{y}\|_2^2}{\mathrm{prox}}(\mathbf{z}) = (\mathbf{I} + \tau \mathbf{A}^* \mathbf{A})^{-1}(\mathbf{z} + \tau \mathbf{A}^* \mathbf{y}). \quad (6)$$

Second, the proximal operator of the ℓ_{21}-norm is

$$\underset{\tau\|[\cdot,\ldots,\cdot]\|_{21}}{\mathrm{prox}}(\mathbf{X}) = \mathrm{soft}_\tau^{\mathrm{row}}(\mathbf{X}), \quad (7)$$

mapping matrix $\mathbf{X} = [x_{ij}]$ to another. It can be shown that it is a soft thresholding over groups consisting of matrix' rows; specifically, it is a mapping

$$x_{ij} \mapsto \frac{x_{ij}}{\|[x_{i1}, \ldots, x_{ik}]\|_2} \max(\|[x_{i1}, \ldots, x_{ik}]\|_2 - \tau). \quad (8)$$

C. Forward-backward based primal-dual splitting

This primal-dual algorithm is actually able to solve more general problems than (4), but we used it for our purpose. For FBB-PD, f_1 must be differentiable with a finite Lipschitz constant. Each iteration of the algorithm consists of two main steps: The "forward step" is a gradient step with respect to f_1 and "backward step" is a proximal step with respect to f_2. These computations are combined with application of L and $L^\top$.

We assign $f_1(\mathbf{x}) = \frac{1}{2}\|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2$. The gradient of f_1 is $\mathbf{A}^\top(\mathbf{A}\cdot - \mathbf{y})$. To guarantee convergence, the Lipschitz constant β plays a crucial role. Value of β can be shown to equal the operator/spectral norm of $\mathbf{A}$, and at the same time, $\|\mathbf{A}\|^2 = \|\mathbf{A}\mathbf{A}^\top\|$. Since rows of $\mathbf{A}$ are mutually orthogonal, $\mathbf{A}\mathbf{A}^\top$ is diagonal with the largest element $1 + N^2 + N^4 + \ldots + N^{2k}$ in the lower-right corner. We apply the summation formula for geometric progression and arrive at the desired constant $\beta = \frac{1 - N^{2(k+1)}}{1 - N^2}$.

We assign $f_2(\mathbf{u}) = f_2(\mathbf{u}_0, \ldots, \mathbf{u}_k) = \|[\mathbf{u}_0, \ldots, \mathbf{u}_k]\|_{21}$. It is a nonsmooth function and the proximal operator of f_2 has been established in (7) and (8). There we have $\tau = 1$.

The linear operator L and its transpose are

$$L(\mathbf{x}) = L(\mathbf{x}_0, \ldots, \mathbf{x}_k) = [\tau_0 \nabla \mathbf{x}_0, \ldots, \tau_k \nabla \mathbf{x}_k] \quad (9)$$

$$L^\top(\mathbf{u}) = L^\top(\mathbf{u}_0, \ldots, \mathbf{u}_k) = \begin{bmatrix} \tau_0 \nabla^\top \mathbf{u}_0 \\ \vdots \\ \tau_k \nabla^\top \mathbf{u}_k \end{bmatrix} \quad (10)$$

with ∇ of size $(N-1)\times N$. The reader may notice that the mapping $L:\mathbb{R}^{kN}\to\mathbb{R}^{N-1\times k}$ also involves reshaping a long vector to a matrix and back, which is needed for the application of the ℓ_{21} norm. The analogue holds for $L^\top:\mathbb{R}^{N-1\times k}\to\mathbb{R}^{kN}$. Operator $\nabla^\top$ can be implemented as efficiently as ∇ since

$$\nabla^\top \mathbf{u} = \nabla\left(\begin{bmatrix}0\\-\mathbf{u}\\0\end{bmatrix}\right). \tag{11}$$

Let us upperbound $\|L\|$ at this moment:

$$\begin{aligned}
\|L\|^2 &= \max_{\|\mathbf{x}\|_2=1}\|L\mathbf{x}\|_2^2\\
&= \max_{\|\mathbf{x}\|_2=1}\|[\tau_0\nabla\mathbf{x}_0,\ldots,\tau_k\nabla\mathbf{x}_k]\|_2^2\\
&= \max_{\|\mathbf{x}\|_2=1}\left(\|\tau_0\nabla\mathbf{x}_0\|_2^2+\ldots+\|\tau_k\nabla\mathbf{x}_k\|_2^2\right)\\
&\le \tau_0^2\max_{\|\mathbf{x}\|_2=1}\|\nabla\mathbf{x}_0\|_2^2+\ldots+\tau_k^2\max_{\|\mathbf{x}\|_2=1}\|\nabla\mathbf{x}_k\|_2^2\\
&\le \tau_0^2\|\nabla\|^2+\ldots+\tau_k^2\|\nabla\|^2\\
&\le 4(\tau_0^2+\ldots+\tau_k^2).
\end{aligned}$$

since it can be easily shown that $\|\nabla\|\le 2$. From here it follows that $\|L\|\le 2\sqrt{\sum_{j=0}^k\tau_j^2}=:2\|\boldsymbol{\tau}\|_2$.

Finally, there is a third function involved in the FBB-PD algorithm, and we assign $f_3=0$.

The main steps of each iteration recompute primal and dual variables by

- $\mathbf{p}^{(n)}=\operatorname{prox}_{\zeta f_3}\left(\mathbf{x}^{(n)}-\zeta(\nabla f_1(\mathbf{x}^{(n)})+\sigma L^\top\mathbf{v}^{(n)})\right)$
- $\mathbf{q}^{(n)}=(Id-\operatorname{prox}_{f_2/\sigma})(\mathbf{v}^{(n)}+L(2\mathbf{p}^{(n)}-\mathbf{x}^{(n)}))$

where ∇ stands for gradient for this paragraph only and ζ,σ are positive scalars called "time steps". In our case, these computations render as

- $\mathbf{p}^{(n)}=\mathbf{x}^{(n)}-\zeta\mathbf{A}^\top(\mathbf{A}\mathbf{x}^{(n)}-\mathbf{y})-\zeta\sigma L^\top\mathbf{v}^{(n)}$
- $\mathbf{q}^{(n)}=(Id-\operatorname{soft}^{\text{row}}_{1/\sigma})(\mathbf{v}^{(n)}+L(2\mathbf{p}^{(n)}-\mathbf{x}^{(n)}))$.

To ensure convergence, the following must be satisfied [19]:

$$1/\zeta\ge\beta/2+\sigma\|L\|^2 \tag{12}$$

and while β is fixed and $\|L\|\le 2\|\boldsymbol{\tau}\|_2$, we have only a single degree of freedom to choose σ,ζ which influence the speed of convergence.

D. Chambolle-Pock splitting

Chambolle-Pock algorithm is tailored to solve problems of type (4). In this iterative algorithm, each iteration consists of two proximal steps with respect to f_1 and f_2, respectively.

Functions f_1 and f_2 are specified in the same way as above. The main difference between FBB-PD and CHP algorithm is that the latter uses the proximal operator of f_1 instead of its gradient; the proximal operator of f_1 is displayed in (6).

The CHP algorithm uses these main steps:

- $\mathbf{q}^{(n+1)}=\operatorname{prox}_{\sigma f_2^*}(\mathbf{q}^{(n+1)}+\sigma L\bar{\mathbf{p}}^{(n+1)})$
- $\mathbf{p}^{(n+1)}=\operatorname{prox}_{\zeta f_1}(\mathbf{p}^{(n+1)}-\zeta L^\top\mathbf{q}^{(n+1)})$

where $\operatorname{prox}_{f^*}$ can be computed at virtually the same cost as prox_f thanks to the Moreau identity [19]. The convergence is ensured when $\zeta\sigma\|L\|^2<1$.

E. Signal segmentation and final estimation

Achieved optimizers $\hat{\mathbf{x}}$ of our problem (3) allow estimation of the underlying signal $\hat{\mathbf{y}}$ according to $\hat{\mathbf{y}}=\mathbf{A}\hat{\mathbf{x}}$. In this section we discuss the means by which this simple reconstruction could be improved at a negligible computational cost.

Nonzero values in $\nabla\hat{\mathbf{x}}_0,\ldots,\nabla\hat{\mathbf{x}}_k$ indicate segment borders. However, only in rare cases, regularization weights $\tau_0,\ldots,\tau_k$ can be precisely tuned to achieve piecewise constant optimizers. The nonstrict convexity of the TV-like regularizer in (3) makes the problem no easier, see, for example, the discussion of such a phenomenon in [15], [21]. Even when τ_js are set properly, vectors $\nabla\hat{\mathbf{x}}_j$ can be full of small values, besides larger values indicating possible segment borders. Recall that the underlying signal is assumed to be piecewise polynomial; therefore, we apply a two-part procedure to obtain the denoised signal: first, borders of the signal segments are detected, and second, optimal parameters are found for each detected segment individually.

The process of changepoints detection tries to avoid false detection caused by small values in the difference vectors. Thanks to the use of mixed norm, significant values in $\nabla\hat{\mathbf{x}}_j$s are situated at the same positions, and our detection procedure can benefit from this property. Thus, we start by making a single vector out of all achieved $\nabla\hat{\mathbf{x}}_j$s, which is done by computing the euclidean distance according to the formula

$$\mathbf{d}=\sqrt{(\alpha_0\nabla\hat{\mathbf{x}}_0)^2+\cdots+(\alpha_k\nabla\hat{\mathbf{x}}_k)^2}, \tag{13}$$

where all operations are considered as acting entrywise on the vectors. Scalars $\alpha_0,\ldots,\alpha_k$ are positive weights used because the range of values in the parametrization vectors grows with the order of the polynomial they correspond to. We compute each α_j as the ratio of maxima of $|\nabla\hat{\mathbf{x}}_k|$ and $|\nabla\hat{\mathbf{x}}_j|$ respectively, implying $\alpha_k=1$. Then, a moving median filter is applied to $\mathbf{d}$ and the obtained filtered signal is subtracted from $\mathbf{d}$, yielding $\hat{\mathbf{d}}$. This approach leaves significant breakpoints and filters the segments between breakpoints, such that it provides signal with more zero (or close to zero) values in $\hat{\mathbf{d}}$. Indices satisfying the condition $\left|\hat{\mathbf{d}}\right|>\lambda$ with a proper threshold λ then constitute the detected breakpoints.

As the final process, denoising/smoothing is applied on each detected segment. This is done simply by forming a regression matrix whose columns are constant, linear, quadratic etc. time vectors, and performing an ordinary least squares method on data $\mathbf{y}$ restricted to the segment range. By doing this, we obtain a new parametrization coefficient set $\dot{\mathbf{x}}$ which is constant on the segment-by-segment basis. The denoised signal $\dot{\mathbf{y}}$ is then reconstructed according to $\dot{\mathbf{y}}=\mathbf{A}\dot{\mathbf{x}}$. The least squares refit could be seen as a debiasing, commonly used in LASSO-type estimation [22], which is inherently biased by the use of the ℓ_1-norm.

IV. Experiments

The experiments have been done in Matlab, and for the proximal algorithms we benefited from using the flexible UnLocBox toolbox[1] [23].

[1] https://lts2.epfl.ch/unlocbox

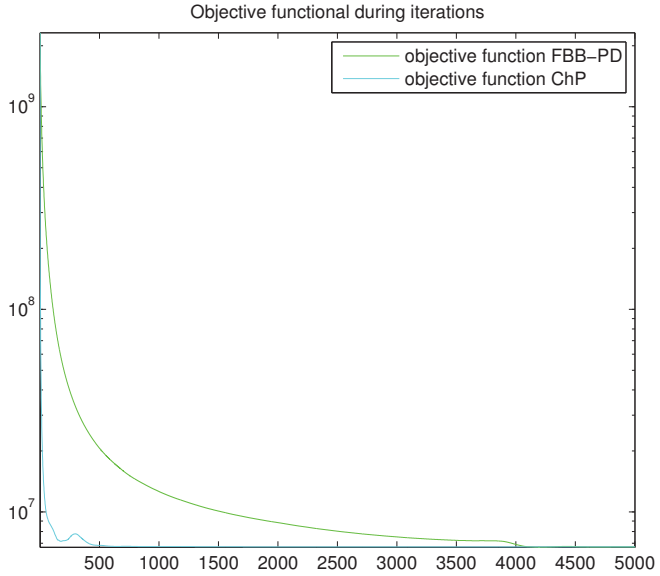

Figure 1. Typical convergence profile of FBB-PD and CHP algorithms for our problem. The value of the objective is plot against the number of iterations. It is obvious that FBB-PD algorithm needs about ten times more iterations to converge in contrast to the CHP algorithm.

A. Convergence

We tested both the FBB-PD and CHP algorithms in solving the problem (3). In all our experiments, the convergence of the CHP algorithm was much faster than of FBB-PD (see Fig. 1). This can be explained by the fact that the gradient step used in FBB-PD is very inefficient since the gradient is dominated by $\mathbf{A}^\top\mathbf{A}$ which is very badly conditioned (recall that the ratio of the largest and the smallest eigenvalue is in the order of N^{2k}, which can get huge easily).

At the same time, the computational cost of a single iteration is comparable between the two algorithms. Indeed, when we return to the main computation steps presented in Sections III-C and III-D and omit vector additions and subtractions, we see that both the algorithms exploit identical operations. To be specific, each iteration performs application of L and $L^\top$, and computes $\mathrm{soft}^{\mathrm{row}}(\cdot)$. The only prominent difference is that FBB-PD needs to evaluate $\mathbf{A}^\top(\mathbf{A}\cdot - \mathbf{y})$, while CHP applies an inverse matrix: $(\mathbf{I}+\zeta\mathbf{A}^\top\mathbf{A})^{-1}(\cdot + \zeta\mathbf{A}^\top\mathbf{y})$. At the first sight, this looks like a huge difference, but consider that $\mathbf{A}^\top\mathbf{A}$ as well as $\mathbf{A}^\top\mathbf{y}$ can be precomputed, and since ζ is a fixed parameter, the matrix inversion could be precomputed, too. This keeps both the algorithms at the same computational order per iteration. This fact was evident also in our numerical experiments.

We note that profiles in Fig. 1 have been produced using default time-steps for both algorithms. Adjusting the time-steps σ, ζ in the FBB-PD algorithm brought a small speedup in the decrease of the objective, but the resulting speed has still been far away from that of CHP. Moreover, we were able to slightly lower the profile of the CHP algorithm by tuning its time-steps.

The reader may notice that the matrix $\mathbf{A}^\top\mathbf{A}$, as well as $(\mathbf{I}+\zeta\mathbf{A}^\top\mathbf{A})$, is very sparse. However, we did not observe a significant speedup by forcing Matlab to use the sparse datatype.

B. Polynomial signal

We present an experiment performed on a randomly generated piecewise polynomial signal of degree 2 and of total length $N = 150$. Parametrization vectors $\mathbf{x}_0$, $\mathbf{x}_1$ and $\mathbf{x}_2$ in $\mathbb{R}^N$ used in the simulation are piecewise constant, according to the assumption. In our case, the signal contained six independent polynomial segments. Signal $\mathbf{y}$ is created using $[\mathbf{I}\,\mathbf{D}\,\mathbf{D}^2][\mathbf{x}_0^\top\,\mathbf{x}_1^\top\,\mathbf{x}_2^\top]^\top$ and the iid Gaussian noise is added on top of it.

C. Results

We tuned the regularization parameters τ_0, τ_1, τ_2 to obtain the parametrization coefficients resulting in values as close as possible to the synthetic ones. The results of this process are depicted in Fig. 2. The results after postprocessing (i.e. detecting segments and refitting by least squares) are depicted in Fig. 3. Our experience is that good results require careful tuning of the model parameters. We did this empirically, although note that there is, for example, a rule that the greater variance the noise has, the greater should be the τ_is.

V. Discussion and Conclusion

The experiments show that the presented methodology of signal segmentation/denoising is promising. However, careful tuning of the parameters must be carried out to obtain good results, which makes the method user-demanding.

In [15], we argued that an easier achievement of good results is in fact complicated by the TV regularizer. As we explained in Sec. II, the ℓ_{21}-norm used in this paper resolves the segment borders better than the simple TV, but it still shares the same properties with the ordinary TV (for example its nonstrict convexity in combination with the difference operator ∇).

One could therefore hope for more robust recovery programs by introducing non-convex regularizers, as such force stepwise nature of the parametrization vectors $\mathbf{x}_j$ in a greater degree. Exploring this way by for example using ℓ_p-"norms" with $p < 1$ will be our future goal, which will serve as a comparison to the state-of-the-art method from [3] that uses joint-sparse approach, which is based on a greedy algorithm and thus is nonconvex as well.

One could also think about "mimicking" nonconvexity via a series of convex programs. Such an approach is not new in general, it has been suggested by [24], for example. To be more specific, we suggest to formulate a series of convex problems in the form of (3), but changing the regularizer to entrywise weighted differences instead of applying a common weight τ_j to vector $\nabla\mathbf{x}_j$. And, most importantly, these weights would change adaptively after a suitable number of iterations, based on the intermediate result. The entrywise weights for difference terms would change such that points which are most probably the breakpoints would be gradually assigned lower weight, thus penalized less.

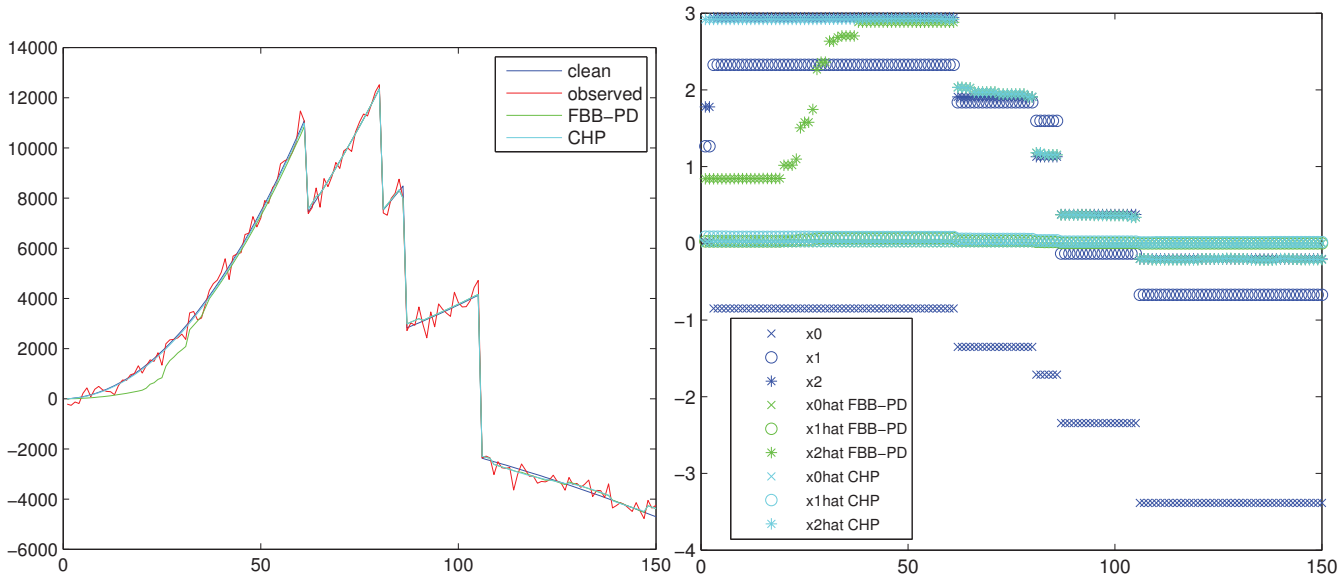

Figure 2. Approximation of a random piecewise polynomial signal with $\tau_0 = 4681100$, $\tau_1 = 5617300$, $\tau_2 = 7021600$ after 1000 iterations. In the figure on the left, we can see the clean signal and its noisy version $\mathbf{y}$, and approximations achieved by FBB-PD and CHP algorithms. In the figure on the right, the respective parametrization coefficients are presented. It is evident that for the FBB-PD algorithm, 1000 iterations was not enough; it did not converge to piecewise solution, while the CHP did. We can also see that the first segment has not been detected by none of the algorithms (or the model (3) strictly speaking); this segment consists of two elements only and it is hardly possible to recognize it in the noisy signal even by eye. The SNR of the input signal was 24.8 dB, SNR of recovered signal after FBB-PD was 24.8 dB and after CHP it was 32.6 dB.

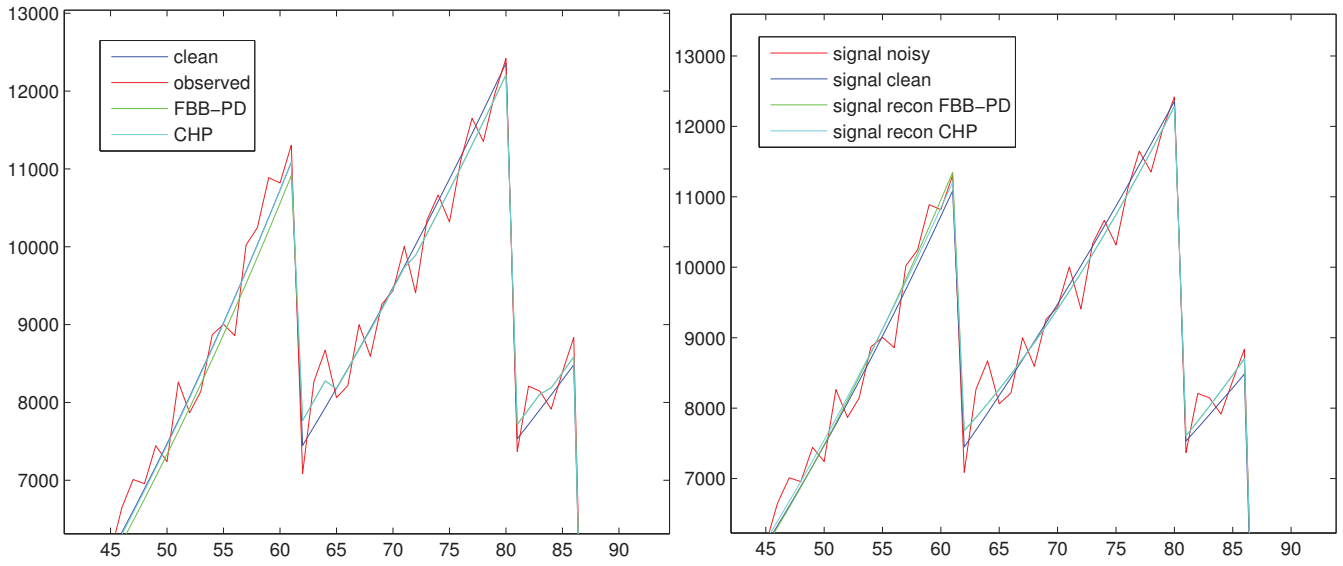

Figure 3. The same signal as in Fig. 2, comparison before and after postprocessing. The left-side figure shows a part of the clean and noisy signal and the approximations by FBB-PD and CHP before postprocessing, i.e. this plot is just a zoom of graph from Fig. 2 left. The figure on the right shows the same situation, but after postprocessing of $\hat{\mathbf{x}}_0, \hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2$ with $\lambda = .5$. In this particular experiment, the automatically computed weights were $\alpha_0 = 719.9, \alpha_1 = 51.3, \alpha_2 = 1$ and $\alpha_0 = 295.6, \alpha_1 = 31.6, \alpha_2 = 1$ for FBB-PD and CHP, respectively. The moving median filter was five samples long. By postprocessing, the best fit to the data in the least squares sense is achieved, segment-by-segment. The SNR of the recovered signal based on FBB-PD was 34.2 dB and based on CHP was 34.8 dB. We see that in two middle segments the FBB-PD and CHP curves coincide, which indicates that the border points of these segments were identified at identical indices. In the leftmost segment, however, the left changepoints for FBB-PD and CHP respectively are not identical and thus the least squares fits differ slightly.

ACKNOWLEDGMENT

The authors want to thank Nathanaël Perraudin, Michal Šorel and Zdeněk Průša for valuable discussion and comments. Research described in this paper was financed by the National Sustainability Program under grant LO1401 and by the Czech Science Foundation under grant no. GA16-13830S. For the research, infrastructure of the SIX Center was used.

REFERENCES

[1] J.-L. Starck, E. J. Candes, and D. L. Donoho, "The curvelet transform for image denoising," *IEEE Transactions on Image Processing*, vol. 11, no. 6, pp. 670–684, Jun 2002.

[2] G. Kutyniok and W.-Q. Lim, "Compactly supported shearlets are optimally sparse," *Journal of Approximation Theory*, vol. 163, no. 11, pp. 1564–1589, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0021904511001067

[3] R. Giryes, M. Elad, and A. Bruckstein, "Sparsity based methods for overparameterized variational problems," *SIAM journal on imaging sciences*, vol. 8, no. 3, pp. 2133–2159, 2015.

[4] J. Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Transactions on Information Theory*, vol. 50, pp. 2231–2242, 2004.

[5] M. Elad, *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*. Springer, 2010.

[6] J. Neubauer and V. Veselý, "Change point detection by sparse parameter estimation," *INFORMATICA*, vol. 22, no. 1, pp. 149–164, 2011.

[7] K. Bleakley and J.-P. Vert, "The group fused Lasso for multiple change-point detection," Jun. 2011, technical report. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00602121

[8] J. Frecon, N. Pustelnik, P. Abry, and L. Condat, "On-the-fly approximation of multivariate total variation minimization," *IEEE Transactions on Signal Processing*, vol. PP, no. 99, pp. 1–1, 2016.

[9] B. Zhang, J. Geng, and L. Lai, "Multiple change-points estimation in linear regression models via sparse group lasso," *IEEE Transactions on Signal Processing*, vol. 63, no. 9, pp. 2209–2224, May 2015.

[10] D. Angelosante and G. B. Giannakis, "Group Lassoing change-points in piecewise-constant AR processes," *EURASIP Journal on Advances in Signal Processing*, vol. 2012, no. 1, pp. 1–16, 2012.

[11] I. W. Selesnick, S. Arnold, and V. R. Dantham, "Polynomial smoothing of time series with additive step discontinuities," *IEEE Transactions on Signal Processing*, vol. 60, no. 12, pp. 6305–6318, Dec 2012.

[12] S.-J. Kim, K. Koh, S. Boyd, and D. Gorinevsky, "ℓ_1 trend filtering," *SIAM Review*, vol. 51, no. 2, pp. 339–360, 2009. [Online]. Available: http://dx.doi.org/10.1137/070690274

[13] R. J. Tibshirani, "Adaptive piecewise polynomial estimation via trend filtering," *Annals of Statistics*, vol. 42, no. 1, pp. 285–323, 2014.

[14] S. Shem-Tov, G. Rosman, G. Adiv, R. Kimmel, and A. M. Bruckstein, *Innovations for Shape Analysis*, ser. Mathematics and Visualization. Springer, 2012, ch. On Globally Optimal Local Modeling: From Moving Least Squares to Over-parametrization, pp. 379–405.

[15] P. Rajmic and M. Novosadová, "On the limitation of convex optimization for sparse signal segmentation," in *Proceedings of the 9th International Conference on Telecommunications and Signal Processing*. Brno University of Technology, 2016, pp. 550–554.

[16] M. Kowalski and B. Torrésani, "Structured Sparsity: from Mixed Norms to Structured Shrinkage," in *SPARS'09 – Signal Processing with Adaptive Sparse Structured Representations*, R. Gribonval, Ed. Inria Rennes – Bretagne Atlantique, 2009, pp. 1–6. [Online]. Available: http://hal.inria.fr/inria-00369577/en/

[17] P. Combettes and J. Pesquet, "Proximal splitting methods in signal processing," *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pp. 185–212, 2011.

[18] A. Chambolle and T. Pock, "A first-order primal-dual algorithm for convex problems with applications to imaging," *Journal of Mathematical Imaging and Vision*, vol. 40, no. 1, pp. 120–145, 2011.

[19] N. Komodakis and J. Pesquet, "Playing with duality: An overview of recent primal-dual approaches for solving large-scale optimization problems," *IEEE Signal Processing Magazine*, vol. 32, no. 6, pp. 31–54, Nov 2015.

[20] L. Condat, "A generic proximal algorithm for convex optimization—application to total variation minimization," *Signal Processing Letters, IEEE*, vol. 21, no. 8, pp. 985–989, Aug 2014.

[21] T. Pock, "Fast total variation for computer vision," Dissertation thesis, Graz University of Technology, 2008.

[22] R. Tibshirani, "Regression shrinkage and selection via the LASSO," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[23] N. Perraudin, D. I. Shuman, G. Puy, and P. Vandergheynst, "Unlocbox A Matlab convex optimization toolbox using proximal splitting methods," 2014. [Online]. Available: http://arxiv.org/abs/1402.0779

[24] E. J. Candes, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *Journal of Fourier Analysis and Applications*, vol. 14, pp. 877–905, 12 2008.