Piecewise-polynomial signal segmentation using proximal splitting convex optimization methods

Michaela Novosadová

Brno University of Technology

June 10, 2016, Applied mathematical programming and Modelling APMOD 2016

Introduction

- Use of convex optimization for signal segmentation/denoising
- Proximal splitting algorithm

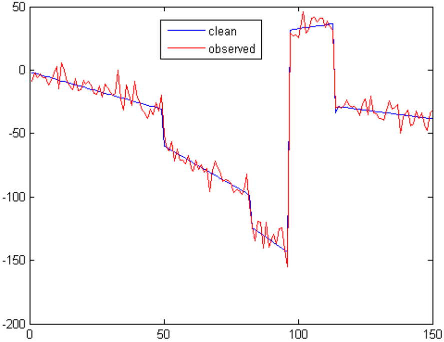
Problem formulation

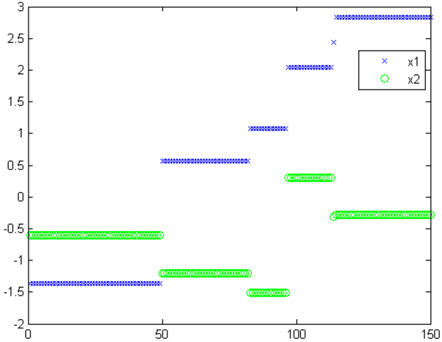
- Overparameterization model
- 1D piecewise polynomial signal $\mathbf{f} \in \mathbb{R}^N$,
  $f(i) = x_1(i) + ix_2(i) \cdots + i^{k-1}x_k(i)$

$$\mathbf{f} = \mathbf{A}\mathbf{x} = [\mathbf{I}\,\mathbf{D}^1, \ldots, \mathbf{D}^{k-1}] \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_k \end{bmatrix} \quad (1)$$

- where N is signal length,
- $\mathbf{x}_1, \ldots, \mathbf{x}_k \in \mathbb{R}^N$, e.g. $\mathbf{x}_1$ is constant offset, $\mathbf{x}_2$ is constant slope etc.,
- $\mathbf{I} = \mathbf{I}_N$ is identity matrix,
- $\mathbf{D} = \mathrm{diag}(1, 2, \ldots, N)$ is diagonal matrix

Problem formulation





$$f(i) = x_1(i) + i x_2(i)$$

Problem formulation

- Recovery problem

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \ \frac{1}{2}\|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \|[\tau_1 \nabla \mathbf{x}_1, \ldots, \tau_k \nabla \mathbf{x}_k]\|_{21} \qquad (2)$$

- where $\mathbf{y} \in \mathbb{R}^N$ is the observed signal, $\mathbf{y} = \mathbf{f} + \mathbf{e}$
- $\mathbf{e} \in \mathbb{R}^N$ is Gaussian noise
- ∇ is the difference operation
- $\tau_1, \ldots, \tau_k$ are regularization weights
- $\hat{\mathbf{x}} = \begin{bmatrix} \hat{\mathbf{x}}_1 \\ \vdots \\ \hat{\mathbf{x}}_k \end{bmatrix}$ are achieved optimizers,
- l_{21} norm - to promote sparsity across groups and not within groups

Methodology

- Signal segmentation — detection of breakpoints
- Signal denoising — ordinary Least Squares Method

Methodology — Proximal algorithms

- Optimization problem

$$\arg\min_{\mathbf{x}} \; f_1(\mathbf{x}) + f_2(L\mathbf{x}) \tag{3}$$

  for convex f_1, f_2 and L a linear operator
- can be solved by *proximal splitting algorithms*, which iteratively act on f_1 and f_2 separately
- for example
  - Forward-Backward
  - Chambolle-Pock
- In our case,
  - $f_1(\mathbf{x}) = \frac{1}{2}\|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2$
  - $f_2(\mathbf{x}) = f_2(\mathbf{x}_1, \ldots, \mathbf{x}_k) = \|[\mathbf{x}_1, \ldots, \mathbf{x}_k]\|_{21}$
  - $L\mathbf{x} = L(\mathbf{x}_1, \ldots, \mathbf{x}_k) = [\tau_1 \nabla \mathbf{x}_1, \ldots, \tau_k \nabla \mathbf{x}_k]$

Methodology — Chambolle-Pock Algorithm

Input: $f_1, f_2, L \in \mathbb{R}^{m \times n}$

- Choose $\tau, \sigma > 0$, $\theta \in [0,1]$
- Choose starting points $\mathbf{p}_0 \in \mathbb{R}^n$, $\mathbf{q}_0 \in \mathbb{R}^m$
- Set $\bar{\mathbf{p}}_0 = \mathbf{p}_0$.
- Iterate $n = 0, 1, 2, \ldots$ until convergence
  - $\mathbf{q}_{n+1} = \operatorname{prox}_{f_2^*}(\mathbf{q}_n + \sigma L \bar{\mathbf{p}}_n)$
  - $\mathbf{p}_{n+1} = \operatorname{prox}_{f_1}(\mathbf{p}_n - \tau L^T \mathbf{q}_{n+1})$
  - $\bar{\mathbf{p}}_{n+1} = \mathbf{p}_{n+1} + \theta(\mathbf{p}_{n+1} - \mathbf{p}_n)$

Methodology — Breakpoints detection

- From solution $\hat{\mathbf{x}}$ compute Euclidean distance

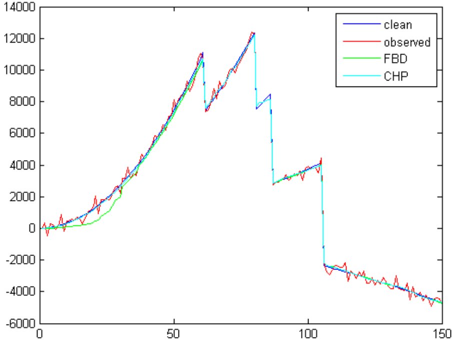
$$\mathbf{b} = \sqrt{\hat{\mathbf{x}}_1^2 + \cdots + \hat{\mathbf{x}}_k^2} \tag{4}$$

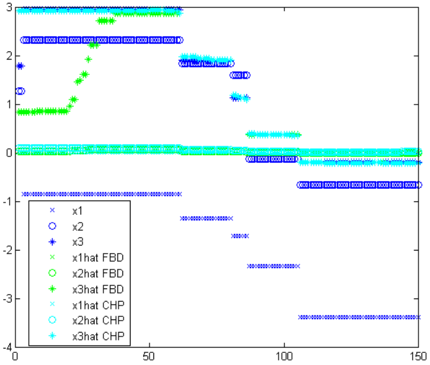
- Breakpoints detection in $\nabla \mathbf{b}$
- Nonzero values in $\nabla \mathbf{b}$ indicate possible segment borders
- Difficult to set regularization weights
- Thresholding of $\nabla \mathbf{b}$ with threshold λ
- Vector of breakpoints positions $\mathbf{bp} = [1, \mathbf{bp}_b, N]$ $\mathbf{bp}_b$ - positions of nonzero values in $\nabla \mathbf{b}$ satisfying condition $|(\nabla \mathbf{b})_i| > \lambda$

Methodology — Signal Denoising

- For each detected segment — least squares method
- $\dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{x}}_1 \\ \vdots \\ \dot{\mathbf{x}}_k \end{bmatrix}$ - vector of new parameters of each segment
- Denoised signal $\hat{\mathbf{y}}$

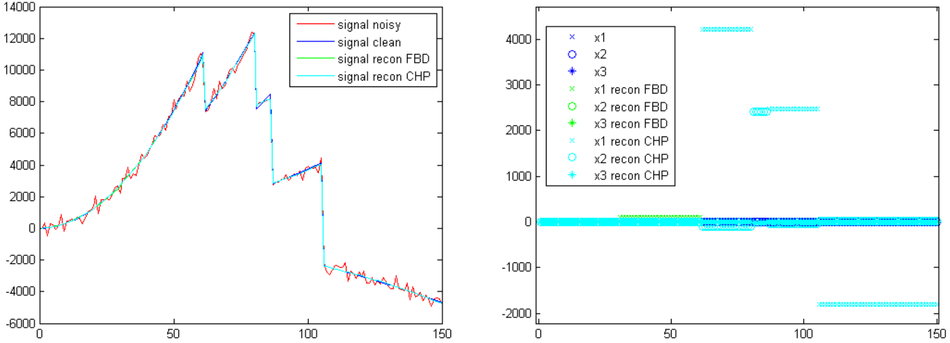
$$\hat{\mathbf{y}} = \mathbf{A}\dot{\mathbf{x}} \tag{5}$$





$\tau_{x1} = 4681100$, $\tau_{x2} = 5617300$, $\tau_{x3} = 7021600$, SNR of observed signal is $SNR = 25.6dB$, recovered signal after first step has $SNR_{FBD} = 25.3dB$, $SNR_{CHP} = 35.1dB$

Experiments — Randomly generated signal: Second step



$\lambda = 0.5$, SNR of observed signal is $SNR = 25.6dB$, recovered signal after second step has $SNR_{FBD} = 39.7dB$, $SNR_{CHP} = 40.1dB$

Conclusion

- Succesful
- Future work: online reweighting differences

Thanks for your attention