COMPRESSIVE ONLINE ROBUST PRINCIPAL COMPONENT ANALYSIS WITH MULTIPLE PRIOR INFORMATION

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Huynh Van Luong[1], Nikos Deligiannis[2], Jürgen Seiler[1], André Kaup[1], and Søren Forchhammer[3]
[1]Multimedia Communications and Signal Processing, University of Erlangen-Nuremberg, 91058 Erlangen, Germany
[2]Vrije Universiteit Brussel, 1050 Brussels, Belgium and imec, Kapeldreef 75, B-3001 Leuven, Belgium
[3]DTU Fotonik, Technical University of Denmark, 2800 Lyngby, Denmark

1. Motivation

- Applications: Computer vision, web data analysis, anomaly detection, and data visualization, etc.
- Robust Principal Component Analysis (RPCA): Batch-based, decomposes all data samples (matrix $\boldsymbol{M}$) into low-rank ($\boldsymbol{L}$) and sparse ($\boldsymbol{S}$), e.g., all frames in a video, high computational and memory requirements

$$\min_{\boldsymbol{L},\boldsymbol{S}} \|\boldsymbol{L}\|_* + \lambda\|\boldsymbol{S}\|_1 \text{ subject to } \boldsymbol{M} = \boldsymbol{L} + \boldsymbol{S}$$

Challenges

- Online method processing a sequence of signals per time instance from a small set of measurements $\boldsymbol{y}_t = \boldsymbol{\Phi}(\boldsymbol{x}_t + \boldsymbol{v}_t)$

$$\boldsymbol{M}_t = \boldsymbol{L}_t + \boldsymbol{S}_t \text{ into } \boldsymbol{S}_t = [\boldsymbol{x}_1\ \boldsymbol{x}_1\ ...\ \boldsymbol{x}_t] \text{ and } \boldsymbol{L}_t = [\boldsymbol{v}_1\ \boldsymbol{v}_2\ ...\ \boldsymbol{v}_t]$$

➢ Minimization at time instance *t*

$$\min_{\boldsymbol{x}_t,\boldsymbol{v}_t} \|[\boldsymbol{L}_{t-1}\ \boldsymbol{v}_t]\|_* + \lambda\|\boldsymbol{x}_t\|_1 \text{ subject to } \boldsymbol{y}_t = \boldsymbol{\Phi}(\boldsymbol{x}_t + \boldsymbol{v}_t)$$

where $\boldsymbol{L}_{t-1} = [\boldsymbol{v}_1\ \boldsymbol{v}_2\ ...\ \boldsymbol{v}_{t-1}]$, $\boldsymbol{S}_{t-1} = [\boldsymbol{x}_1\ \boldsymbol{x}_1\ ...\ \boldsymbol{x}_{t-1}]$, $\boldsymbol{\Phi}$ are given

2. Compressive Online RPCA (CORPCA) With Multiple Prior Information

Problem formulation

- Incorporating multiple prior information: at time instance *t* we observe $\boldsymbol{y}_t = \boldsymbol{\Phi}(\boldsymbol{x}_t + \boldsymbol{v}_t)$ with $\boldsymbol{y}_t \in \mathbb{R}^m$ given priors $\boldsymbol{Z}_{t-1}$ and $\boldsymbol{B}_{t-1}$ from $\{\widehat{\boldsymbol{x}}_1, ..., \widehat{\boldsymbol{x}}_{t-1}\}$ and $\{\widehat{\boldsymbol{v}}_1, ..., \widehat{\boldsymbol{v}}_{t-1}\}$
- Solving the n-ℓ_1 minimization problem

$$\min_{\boldsymbol{x}_t,\boldsymbol{v}_t}\Big\{H(\boldsymbol{x}_t, \boldsymbol{v}_t|\boldsymbol{y}_t, \boldsymbol{Z}_{t-1}, \boldsymbol{B}_{t-1}) = \frac{1}{2}\|\boldsymbol{\Phi}(\boldsymbol{x}_t + \boldsymbol{v}_t) - \boldsymbol{y}_t\|_2^2 + \lambda\mu\sum_{j=0}^{J}\beta_j\|\mathbf{W}_j(\boldsymbol{x}_t - \boldsymbol{z}_j)\|_1 + \mu\Big\|[\boldsymbol{B}_{t-1}\ \boldsymbol{v}_t]\Big\|_*\Big\},$$

where $\lambda > 0$ and $\beta_j > 0$ are weights across the side information signals, and $\mathbf{W}_j$ is a diagonal matrix with weights for each element in the side information signal $\boldsymbol{z}_j$; namely, $\mathbf{W}_j = \mathrm{diag}(w_{j1}, w_{j2}, ..., w_{jn})$ with $w_{ji} > 0$ being the weight for the i-th element in the $\boldsymbol{z}_j$ vector.

The CORPCA algorithm

- Solving n-ℓ_1 minimization via the soft thresholding operator and the single value thresholding operator, at iteration *k + 1*

$$\boldsymbol{v}_t^{(k+1)} = \arg\min_{\boldsymbol{v}_t}\Big\{\mu h(\boldsymbol{v}_t) + \Big\|\boldsymbol{v}_t - \Big(\boldsymbol{v}_t^{(k)} - \frac{1}{2}\nabla_{\boldsymbol{v}_t} f(\boldsymbol{v}_t^{(k)}, \boldsymbol{x}_t^{(k)})\Big)\Big\|_2^2\Big\}$$

$$\boldsymbol{x}_t^{(k+1)} = \arg\min_{\boldsymbol{x}_t}\Big\{\mu g(\boldsymbol{x}_t) + \Big\|\boldsymbol{x}_t - \Big(\boldsymbol{x}_t^{(k)} - \frac{1}{2}\nabla_{\boldsymbol{x}_t} f(\boldsymbol{v}_t^{(k)}, \boldsymbol{x}_t^{(k)})\Big)\Big\|_2^2\Big\}$$

where $f(\boldsymbol{v}_t, \boldsymbol{x}_t) = (1/2)\|\boldsymbol{\Phi}(\boldsymbol{x}_t + \boldsymbol{v}_t) - \boldsymbol{y}_t\|_2^2$
$g(\boldsymbol{x}_t) = \lambda\sum_{j=0}^{J}\beta_j\|\mathbf{W}_j(\boldsymbol{x}_t - \boldsymbol{z}_j)\|_1$, and $h(\boldsymbol{v}_t) = \|[\boldsymbol{B}_{t-1}\ \boldsymbol{v}_t]\|_*$

- Updating weights β_j and $\mathbf{W}_j$
- After solving for time instance *t*: Prior updates

$$\boldsymbol{Z}_t := \{\boldsymbol{z}_j = \boldsymbol{x}_{t-J+j}\}_{j=1}^{J}$$

$$\boldsymbol{B}_t = \boldsymbol{U}_t(:, 1:d)\boldsymbol{\Gamma}_{\frac{\mu_k}{2}g_1}(\boldsymbol{\Sigma}_t)(1:d, 1:d)\boldsymbol{V}_t(:, 1:d)^{\mathrm{T}}$$

Algorithm 1: The proposed CORPCA algorithm.

Input: $\boldsymbol{y}_t, \boldsymbol{\Phi}, \boldsymbol{Z}_{t-1}, \boldsymbol{B}_{t-1}$;
Output: $\widehat{\boldsymbol{x}}_t, \widehat{\boldsymbol{v}}_t, \boldsymbol{Z}_t, \boldsymbol{B}_t$;
// Initialize variables and parameters.
$\boldsymbol{x}_t^{(-1)} = \boldsymbol{x}_t^{(0)} = \boldsymbol{0}$; $\boldsymbol{v}_t^{(-1)} = \boldsymbol{v}_t^{(0)} = \boldsymbol{0}$; $\xi_{-1} = \xi_0 = 1$; $\mu_0 = 0$; $\bar{\mu} > 0$; $\lambda > 0$; $0 < \epsilon < 1$; $k = 0$; $g_1(\cdot) = \|\cdot\|_1$;
while *not converged* **do**
  // Solve Problem (4).
  $\widetilde{\boldsymbol{v}_t}^{(k)} = \boldsymbol{v}_t^{(k)} + \frac{\xi_{k-1}-1}{\xi_k}(\boldsymbol{v}_t^{(k)} - \boldsymbol{v}_t^{(k-1)})$;
  $\widetilde{\boldsymbol{x}_t}^{(k)} = \boldsymbol{x}_t^{(k)} + \frac{\xi_{k-1}-1}{\xi_k}(\boldsymbol{x}_t^{(k)} - \boldsymbol{x}_t^{(k-1)})$;
  $\nabla_{\boldsymbol{v}_t} f(\widetilde{\boldsymbol{v}_t}^{(k)}, \widetilde{\boldsymbol{x}_t}^{(k)}) = \nabla_{\boldsymbol{x}_t} f(\widetilde{\boldsymbol{v}_t}^{(k)}, \widetilde{\boldsymbol{x}_t}^{(k)}) = \boldsymbol{\Phi}^{\mathrm{T}}\Big(\boldsymbol{\Phi}(\widetilde{\boldsymbol{v}_t}^{(k)} + \widetilde{\boldsymbol{x}_t}^{(k)}) - \boldsymbol{y}_t\Big)$;
  $(\boldsymbol{U}_t, \boldsymbol{\Sigma}_t, \boldsymbol{V}_t) = \mathrm{incSVD}\Big(\Big[\boldsymbol{B}_{t-1}\ \Big(\widetilde{\boldsymbol{v}_t}^{(k)} - \frac{1}{2}\nabla_{\boldsymbol{v}_t} f(\widetilde{\boldsymbol{v}_t}^{(k)}, \widetilde{\boldsymbol{x}_t}^{(k)})\Big)\Big]\Big)$;
  $\boldsymbol{\Theta}_t = \boldsymbol{U}_t\boldsymbol{\Gamma}_{\frac{\mu_k}{2}g_1}(\boldsymbol{\Sigma}_t)\boldsymbol{V}_t^T$;
  $\boldsymbol{v}_t^{(k+1)} = \boldsymbol{\Theta}_t(:, \mathrm{end})$;
  $\boldsymbol{x}_t^{(k+1)} = \Gamma_{\frac{\mu_k}{2}g}\Big(\widetilde{\boldsymbol{x}_t}^{(k)} - \frac{1}{2}\nabla_{\boldsymbol{x}_t} f(\widetilde{\boldsymbol{v}_t}^{(k)}, \widetilde{\boldsymbol{x}_t}^{(k)})\Big)$; where $\Gamma_{\frac{\mu_k}{2}g}(\cdot)$ is given as in RAMSIA [18];
  // Compute the updated weights [18].
  $w_{ji} = \frac{n(|x_{ti}^{(k+1)} - z_{ji}| + \epsilon)^{-1}}{\sum_{l=1}^{n}(|x_{tl}^{(k+1)} - z_{jl}| + \epsilon)^{-1}}$;
  $\beta_j = \frac{\Big(\|\mathbf{W}_j(\boldsymbol{x}_t^{(k+1)} - \boldsymbol{z}_j)\|_1 + \epsilon\Big)^{-1}}{\sum_{l=0}^{J}\Big(\|\mathbf{W}_l(\boldsymbol{x}_t^{(k+1)} - \boldsymbol{z}_l)\|_1 + \epsilon\Big)^{-1}}$;
  $\xi_{k+1} = (1 + \sqrt{1 + 4\xi_k^2})/2$; $\mu_{k+1} = \max(\epsilon\mu_k, \bar{\mu})$;
  $k = k + 1$;
end
// Update prior information.
$\boldsymbol{Z}_t := \{\boldsymbol{z}_j = \boldsymbol{x}_{t-J+j}^{(k+1)}\}_{j=1}^{J}$;
$\boldsymbol{B}_t = \boldsymbol{U}_t(:, 1:d)\boldsymbol{\Gamma}_{\frac{\mu_k}{2}g_1}(\boldsymbol{\Sigma}_t)(1:d, 1:d)\boldsymbol{V}_t(:, 1:d)^{\mathrm{T}}$;
return $\widehat{\boldsymbol{x}}_t = \boldsymbol{x}_t^{(k+1)}$, $\widehat{\boldsymbol{v}}_t = \boldsymbol{v}_t^{(k+1)}$, $\boldsymbol{Z}_t, \boldsymbol{B}_t$;

3. Experimental Results

Synthetic data

- Generating low-rank components: *n* = 500, *d* = 100 (training), *q* = 100 (testing), *r* = 5 (rank)
  $\boldsymbol{L} = \boldsymbol{U}\boldsymbol{V}^{\mathrm{T}}$, where $\boldsymbol{U} \in \mathbb{R}^{n\times r}$ and $\boldsymbol{V} \in \mathbb{R}^{(d+q)\times r}$ yields $\boldsymbol{L} = [\boldsymbol{v}_1\ ...\ \boldsymbol{v}_{d+q}]$
- Generating sparse components with $\|\boldsymbol{x}_t\|_0 = s_0$ and $\|\boldsymbol{x}_t - \boldsymbol{x}_{t-1}\|_0 = s_0/2$ obtaining $\boldsymbol{S} = [\boldsymbol{x}_1\ ...\ \boldsymbol{x}_{d+q}]$
- Testing on $\boldsymbol{M} = [\boldsymbol{x}_{d+1} + \boldsymbol{v}_{d+1}\ ...\ \boldsymbol{x}_{d+q} + \boldsymbol{v}_{d+q}]$
- Measuring probabilities of successful decomposition, Pr(success), success if $\|\widehat{\boldsymbol{x}}_t - \boldsymbol{x}_t\|_2/\|\boldsymbol{x}_t\|_2 \leq 10^{-2}$

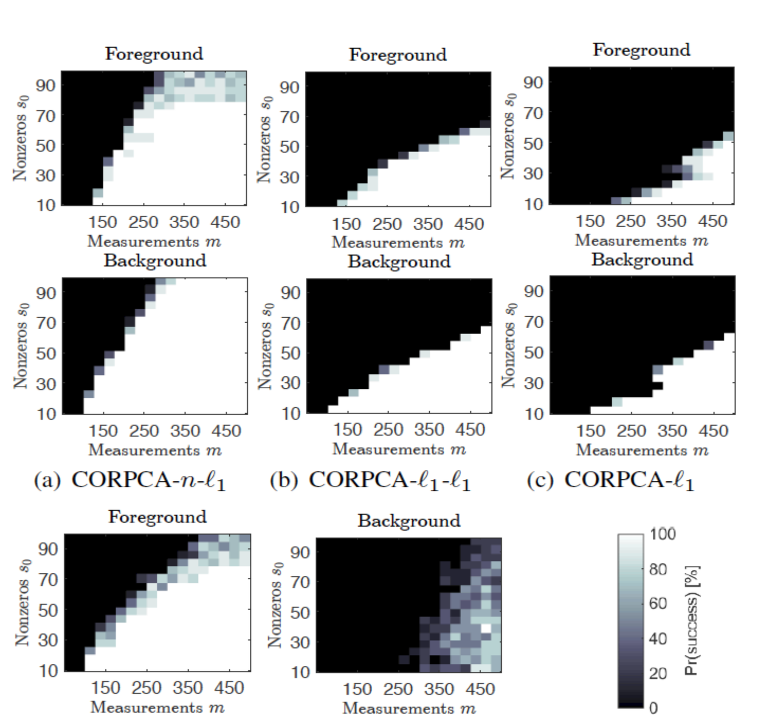

Fig. 1. Average success probabilities for CORPCA (for $\boldsymbol{x}_t$, $\boldsymbol{v}_t$), ReProCS (for $\boldsymbol{x}_t$), and GRASTA (for $\boldsymbol{v}_t$). The scale (f) is proportional to Pr(success)[%] from black to white.

Compressive video foreground-background separation

- Considering two videos, Bootstrap (60x80 pixels) and Curtain (64x80 pixels) having a static and a dynamic background, respectively
- Background-foreground video separation with full access to the video data
- Compressive separating by varying rates on the number of measurements *m* over the dimension of the data *n*

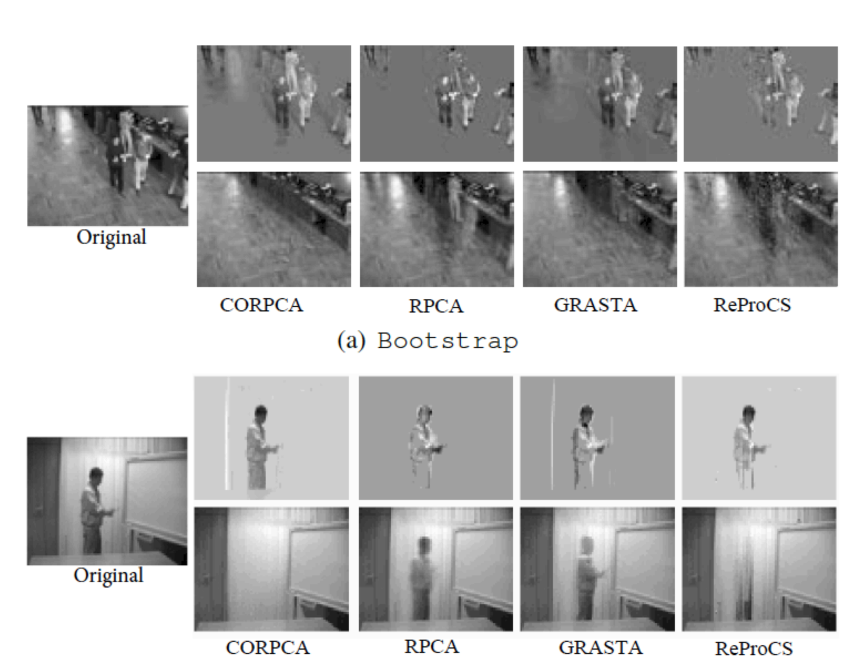

Fig. 2. Background and foreground separation for the different separation methods with full data access Bootstrap#2213 and Curtain#2866.

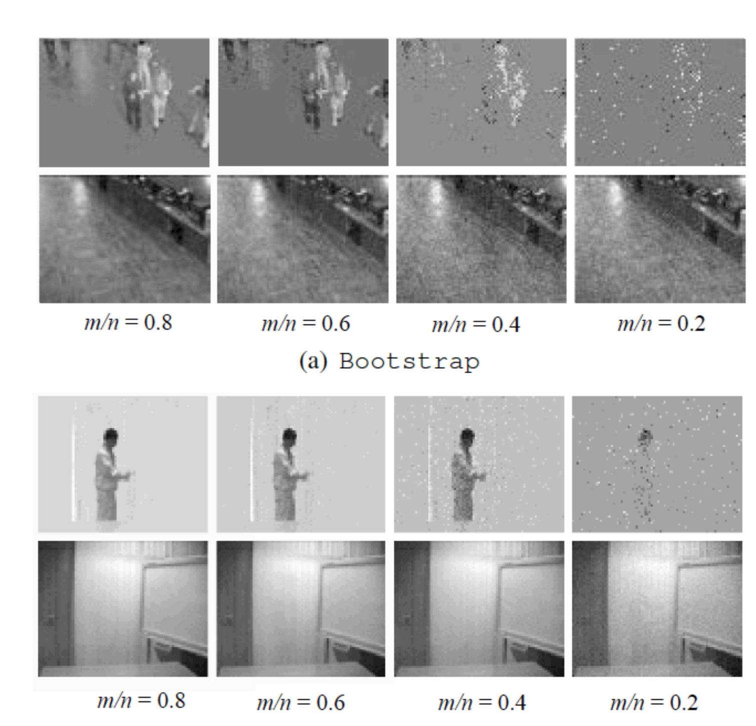

Fig. 3. Compressive background and foreground separation of CORPCA with different measurement rates m/n.

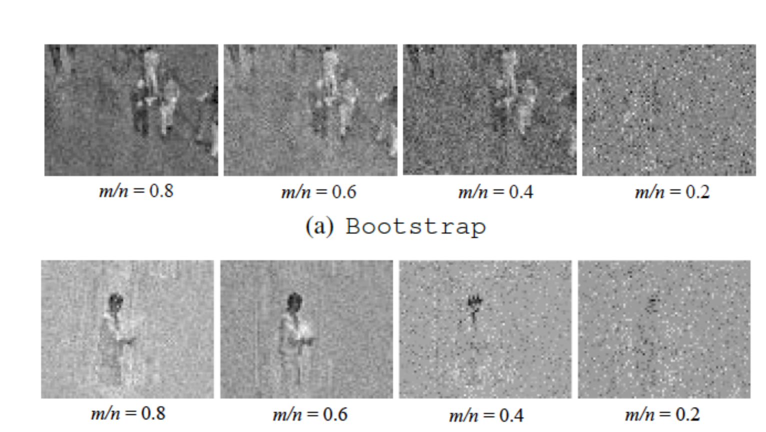

Fig. 4. Compressive foreground separation of ReProCS with different measurement rates m/n.

RPCA: E. J. Candes, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" J. ACM, vol. 58, no. 3, pp. 11:1–11:37, Jun. 2011.
ℓ_1-ℓ_1 minimization: J. F. C. Mota, N. Deligiannis, and M. R. D. Rodrigues, "Compressed sensing with prior information: Strategies, geometry, and bounds," IEEE Trans. Inf. Theory, vol. 63, no. 7, pp. 4472–4496, Jul. 2017.
GRASTA: J. He, L. Balzano, and A. Szlam, "Incremental gradient on the grassmannian for online foreground and background separation in subsampled video," in IEEE Conference on Computer Vision and Pattern Recognition, June 2012.
ReProCS: H. Guo, C. Qiu, and N. Vaswani, "An online algorithm for separating sparse and low-dimensional signal sequences from their sum," IEEE Trans. Signal Process., vol. 62, no. 16, pp. 4284–4297, 2014.

4. Summary

Solution for an n-ℓ_1 minimization

- Incorporating efficiently multiple prior information
- Updating iteratively weights

The proposed COPRCA algorithm

- Processing a data vector per time instance using compressive measurements
- Solving the n-ℓ_1 minimization and updating priors for the next instance

Evaluation of COPRCA on synthetic data and actual video data

- Outperforming classical compressive sensing (CS) (ℓ_1 minimization) and CS with single prior information (ℓ_1-ℓ_1 minimization)
- The superior performance improvement compared to the existing methods

CORPCA source code, test sequences, and the corresponding outcomes. [Online]. Available: https://github.com/huynhlvd/corpca