Fast Automatic Background Extraction via Robust PCA

Ivan Papusha

EE364b: Convex Optimization II Class Project

STANFORD UNIVERSITY

Problem

- Would like to automatically extract background and foreground parts of a sequence of images (2D) or point-cloud frames (3D)
- Generate a data matrix $M \in \mathbf{R}^{m \times n}$, $m \sim 10^5$, $n \sim 10^2$ by concatenating several aligned frames as columns
- Do not know a priori which parts are foreground

Preprocessing

- Align frames (not required for RASL, Peng et.al. 2010)
- Voxel binning (2D or 3D histogram)

Robust PCA

- Approximate $M = L + S$ as a low-rank plus sparse matrix.
  The problem we would like to solve:

$$\begin{array}{ll} \text{minimize} & \mathbf{rank}(L) + \lambda\, \mathbf{card}(S) \\ \text{subject to} & L + S = M, \end{array} \qquad (1)$$

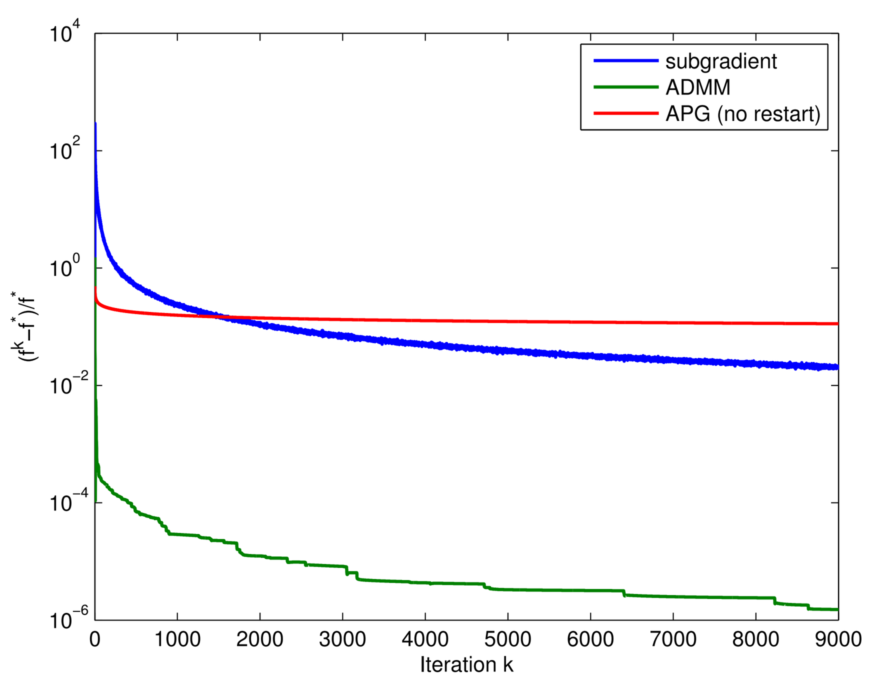
  Principal Component Pursuit (PCP) is convex surrogate:

$$\begin{array}{ll} \text{minimize} & \|L\|_* + \lambda \|S\|_1 \\ \text{subject to} & L + S = M \end{array} \qquad (2)$$

  - $\|L\|_* = \sum_i \sigma_i(L)$ is the nuclear norm
  - $\|S\|_1 = \sum_{i,j} |S_{ij}|$ is ℓ_1-norm of matrix thought of as vector
- parameter $\lambda > 0$ trades off sparsity and rank; fixed value $\lambda = 1/\sqrt{\max(m,n)}$ can lead to *exact reconstruction*! (Candes et.al. 2009)
- Robust to *grossly* corrupted observations of M: single large corruption in an entry of M renders arbitrarily bad estimates of L.
- sparsity pattern of S unknown ahead of time

Typical convergence

- objective vs iteration for small example, $M \in \mathbf{R}^{30 \times 10}$, $M = L_0 + S_0$
- random L_0 with $\mathbf{rank}(L_0) = 3$, random S_0 with 10% nonzero entries



Subgradient Method

- Equivalent problem: equality constraint eliminated,

$$\text{minimize} \;\; f(L) = \|L\|_* + \lambda \|M - L\|_1,$$

  with variable $L \in \mathbf{R}^{m \times n}$. We can recover the optimal sparse term $S^\star$ from the optimal low-rank term $L^\star$ by setting $S^\star = M - L^\star$.
- Calculating a subgradient: $UV^T - \lambda\, \mathbf{sign}(M - L) \in \partial f(L)$, where $L = U \Sigma V^T$ is the singular value decomposition of L.
- Stepsize: $\alpha_k = 1/k^{0.6}$ chosen to be square summable, but not summable.
- **Conclusion**: ...slow and not worth it...but you already knew that!

ADMM

- Solves equality constrained problem over variables $L, S \in \mathbf{R}^{m \times n}$,

$$\begin{array}{ll} \text{minimize} & \|L\|_* + \lambda \|S\|_1 \\ \text{subject to} & L + S = M. \end{array}$$

- Augmented Lagrangian has quadratic penalty on equality constraint

$$\begin{aligned} \mathcal{L}_\rho(L, S, Y) = \|L\|_* + \lambda \|S\|_1 + \mathbf{Tr}(Y^T(L + S - M)) \\ + (\rho/2)\|L + S - M\|_F^2 \end{aligned}$$

- Prox terms given exactly by soft thresholding
  - nuclear norm: singular values (SVD required)
  - ℓ_1-norm: termwise
- **Partial/Randomized SVD** (Drineas et.al. 2004), Lanczos iteration
- Automatic adjustment of penalty parameter ρ gives fast, self contained, parameter free PCP implementation for up to $\mathbf{vec}(M) \in \mathbf{R}^{10^7}$
- Implementation (matlab) available for download, use as simple as `[L,S]=pcp(M)`
- **Conclusion**: ...state of the art!

Accelerated Proximal Gradient

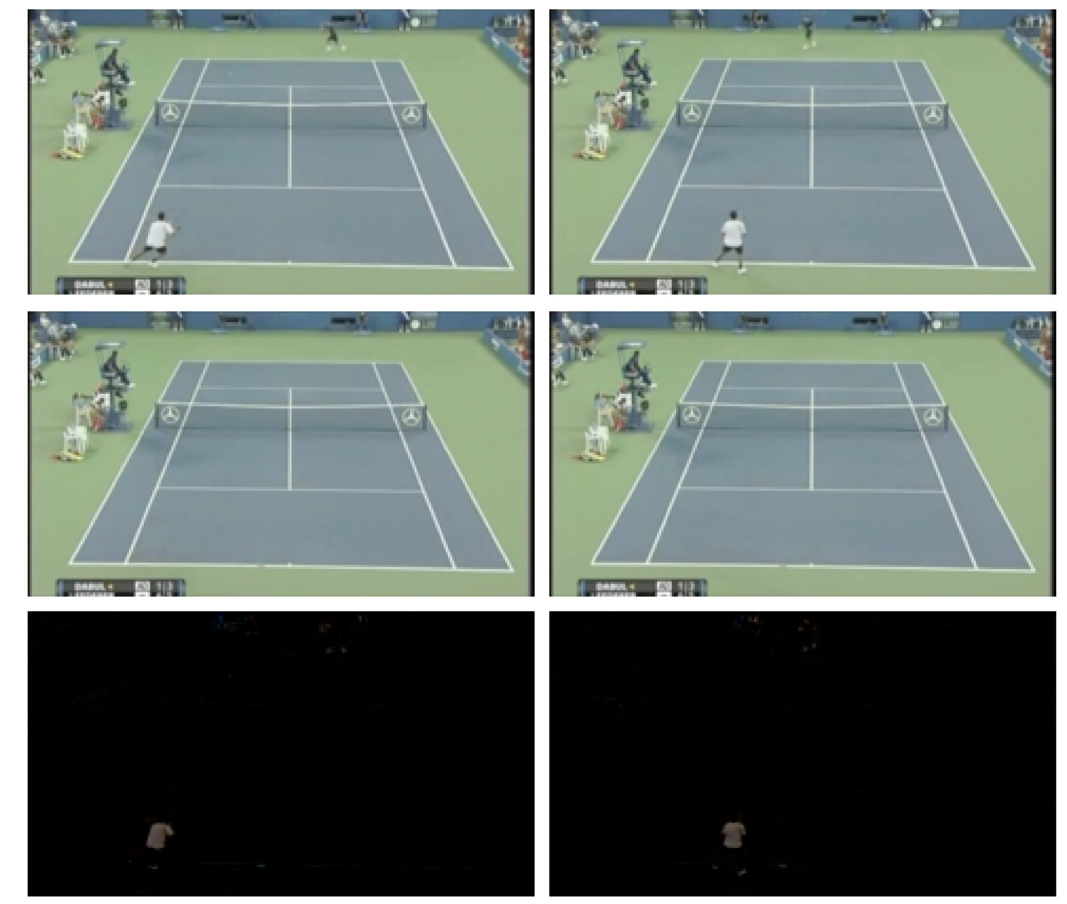
Initialize: $x^{(0)} = y^{(0)} \in \mathbf{dom}\, f$

For $k = 1, 2, \ldots$

1. $x^{(k)} := \mathbf{prox}_{t_k f}(y^{(k-1)}) = \mathrm{argmin}_u \left(f(u) + (1/2t_k)\|u - y^{(k-1)}\|_2^2 \right)$
2. $v^{(k)} := x^{(k-1)} + (1/\theta_k)(x^{(k)} - x^{(k-1)})$
3. $y^{(k)} := (1 - \theta_{k+1}) x^{(k)} + \theta_{k+1} v^{(k)}$

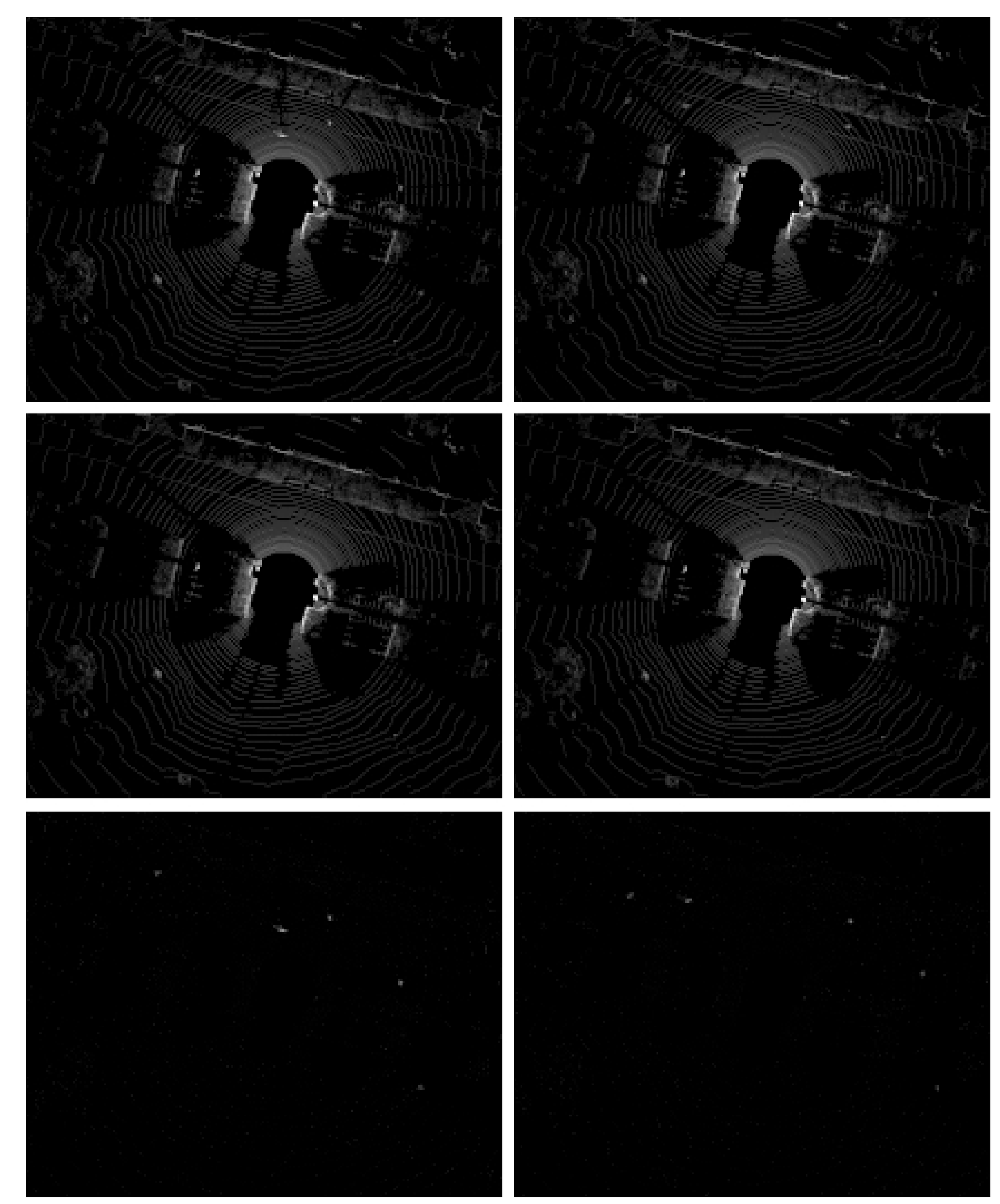
- Proximal gradient with Nesterov-style acceleration (no restarting)
- minimizes non-smooth function $f(L) = \|L\|_* + \lambda \|M - L\|_1$
- Stepsize: $\theta_k = 2/(k+1)$, and $t_k = t$ constant (bounded away from zero)
- **Conclusion**: fragile; results substantiated by literature (Lin et.al. 2009)

Tennis players are sparse!



- $240 \times 135 \times 3$ (color) pixels, 100 frames $\Rightarrow M \in \mathbf{R}^{97200 \times 100}$
- Converges in ~ 40 iterations of ADMM (~ 3 minutes)
- Two frames viewed above (columns)
- First row: original, middle row: low-rank part, bottom row: sparse part

LIDAR mapping



- 214×173 histogram of point cloud, 200 frames $\Rightarrow M \in \mathbf{R}^{37022 \times 200}$
- Same algorithm with no parameter changes!