Robust PCA

CS5240 Theoretical Foundations in Multimedia

Leow Wee Kheng

Department of Computer Science
School of Computing
National University of Singapore

Previously...

not robust against outliers		robust against outliers
linear least squares	➡	trimmed least squares
PCA	➡	trimmed PCA

Various ways to robustify PCA:

- trimming: remove outliers
- covariance matrix with 0-1 weight [Xu95]: similar to trimming
- weighted SVD [Gabriel79]: weighting
- robust error function [Torre2001]: winsorizing

Strength: simple concepts
Weakness: no guarantee of optimality

Robust PCA

PCA can be formulated as follows:

> *Given a data matrix* $\mathbf{D}$*, recover a low-rank matrix* $\mathbf{A}$ *from* $\mathbf{D}$ *such that the error* $\mathbf{E} = \mathbf{D} - \mathbf{A}$ *is minimized:*

$$\min_{\mathbf{A},\mathbf{E}} \|\mathbf{E}\|_F, \text{ subject to } \operatorname{rank}(\mathbf{A}) \leq r, \ \mathbf{D} = \mathbf{A} + \mathbf{E}. \tag{1}$$

- $r \ll \min(m, n)$ is the target rank of $\mathbf{A}$.
- $\| \cdot \|_F$ is the Frobenius norm.

Notes:

- This definition of PCA includes dimensionality reduction.
- PCA is severely affected by large-amplitude noise; not robust.

[Wright2009] formulated the **Robust PCA** problem as follows:

> *Given a data matrix* $\mathbf{D} = \mathbf{A} + \mathbf{E}$ *where* $\mathbf{A}$ *and* $\mathbf{E}$ *are unknown but* $\mathbf{A}$ *is low-rank and* $\mathbf{E}$ *is sparse, recover* $\mathbf{A}$.

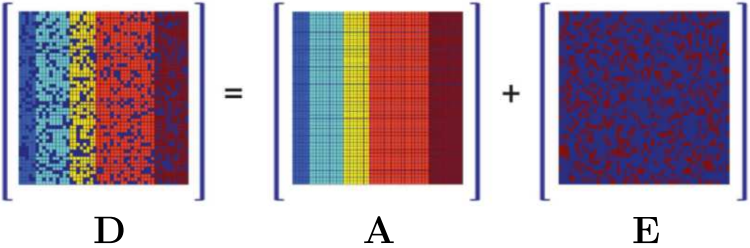
An obvious way to state the robust PCA problem in math is:

> *Given a data matrix* $\mathbf{D}$, *find* $\mathbf{A}$ *and* $\mathbf{E}$ *that solve the problem*

$$\min_{\mathbf{A},\mathbf{E}} \operatorname{rank}(\mathbf{A}) + \lambda \|\mathbf{E}\|_0, \quad \text{subject to } \mathbf{A} + \mathbf{E} = \mathbf{D}. \tag{2}$$

- λ is a Lagrange multiplier.
- $\|\mathbf{E}\|_0$: l_0-norm, number of non-zero elements in $\mathbf{E}$.
  $\mathbf{E}$ is sparse if $\|\mathbf{E}\|_0$ is small.

$$\min_{\mathbf{A},\mathbf{E}} \operatorname{rank}(\mathbf{A}) + \lambda\|\mathbf{E}\|_0, \text{ subject to } \mathbf{A}+\mathbf{E}=\mathbf{D}. \tag{2}$$



- Problem 2 is a **matrix recovery** problem.
- $\operatorname{rank}(\mathbf{A})$ and $\|\mathbf{E}\|_0$ are not continuous, not convex; very hard to solve; no efficient algorithm.

[Candès2011, Wright2009] reformulated Problem 2 as follows:

Given an $m \times n$ data matrix $\mathbf{D}$, find $\mathbf{A}$ and $\mathbf{E}$ that solve

$$\min_{\mathbf{A},\mathbf{E}} \|\mathbf{A}\|_* + \lambda\|\mathbf{E}\|_1, \text{ subject to } \mathbf{A} + \mathbf{E} = \mathbf{D}. \tag{3}$$

- $\|\mathbf{A}\|_*$: nuclear norm, sum of singular values of $\mathbf{A}$; surrogate for $\text{rank}(\mathbf{A})$.
- $\|\mathbf{E}\|_1$: l_1-norm, sum of absolute values of elements of $\mathbf{E}$; surrogate for $\|\mathbf{E}\|_0$.

Solution of Problem 3 can be recovered exactly if

- $\mathbf{A}$ is sufficiently low-rank but not sparse, and
- $\mathbf{E}$ is sufficiently sparse but not low-rank,

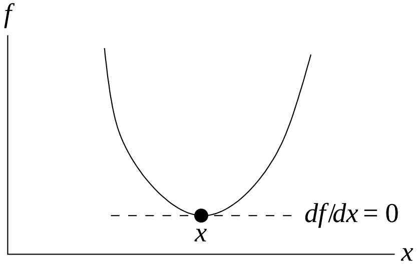
with optimal $\lambda = 1/\sqrt{\max(m, n)}$.

- $\|\mathbf{A}\|_*$ and $\|\mathbf{E}\|_1$ are convex; can apply convex optimization.

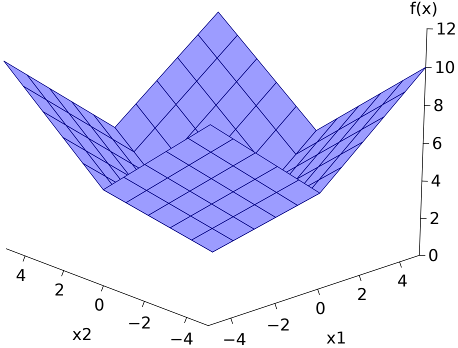
Introduction to Convex Optimization

For a differentiable function $f(\mathbf{x})$, its minimizer $\widehat{\mathbf{x}}$ is given by

$$\frac{df(\widehat{\mathbf{x}})}{d\mathbf{x}} = \left[\frac{\partial f(\widehat{\mathbf{x}})}{\partial x_1} \;\cdots\; \frac{\partial f(\widehat{\mathbf{x}})}{\partial x_m}\right]^\top = \mathbf{0}. \tag{4}$$



$\|\mathbf{E}\|_1$ is not differentiable when any of its element is zero!



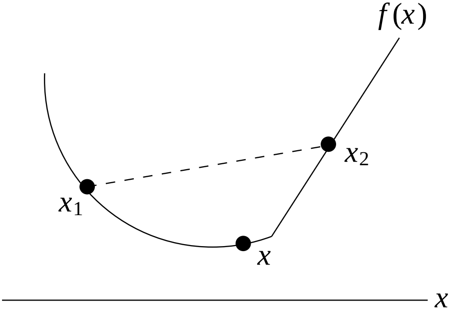
Cannot write the following because they don't exist:

$$\frac{d\|\mathbf{E}\|_1}{d\mathbf{E}}, \quad \frac{\partial\|\mathbf{E}\|_1}{\partial e_{ij}}, \quad \frac{d|e_{ij}|}{de_{ij}}. \quad \textbf{WRONG!} \tag{5}$$

Fortunately, $\|\mathbf{E}\|_1$ is convex.

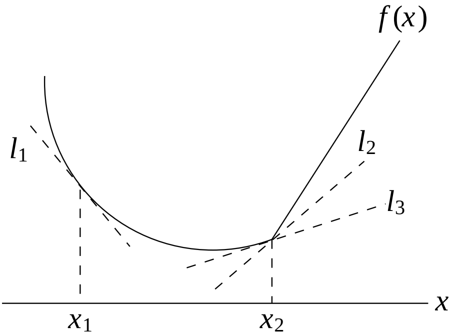
A function $f(\mathbf{x})$ is convex if and only if $\forall \mathbf{x}_1, \mathbf{x}_2$, $\forall \alpha \in [0, 1]$,

$$f(\alpha \mathbf{x}_1 + (1 - \alpha)\mathbf{x}_2) \leq \alpha f(\mathbf{x}_1) + (1 - \alpha) f(\mathbf{x}_2). \tag{6}$$



A vector $\mathbf{g}(\mathbf{x})$ is a subgradient of convex function f at $\mathbf{x}$ if

$$f(\mathbf{y}) - f(\mathbf{x}) \geq \mathbf{g}(\mathbf{x})^\top (\mathbf{y} - \mathbf{x}), \quad \forall \mathbf{y}. \tag{7}$$



- At differentiable point x_1: one unique subgradient = gradient.
- At non-differentiable point x_2: multiple subgradients.

The subdifferential $\partial f(\mathbf{x})$ is the **set** of all subgradients of f at $\mathbf{x}$:

$$\partial f(\mathbf{x}) = \left\{ \mathbf{g}(\mathbf{x}) \,|\, \mathbf{g}(\mathbf{x})^\top (\mathbf{y} - \mathbf{x}) \leq f(\mathbf{y}) - f(\mathbf{x}), \ \forall \mathbf{y} \right\}. \tag{8}$$

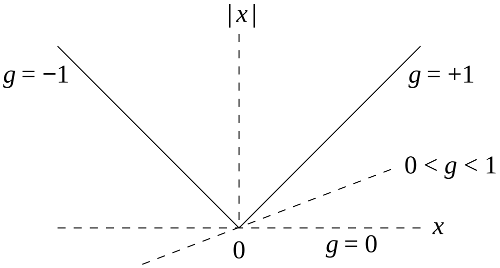
Caution:

- Subdifferential $\partial f(\mathbf{x})$ is **not** partial differentiation $\partial f(\mathbf{x})/\partial \mathbf{x}$. Don't mix up.
- Partial differentiation is defined for differentiable functions. It is a single vector.
- Subdifferential is defined for convex functions, which may be non-differentiable. It is a set of vectors.

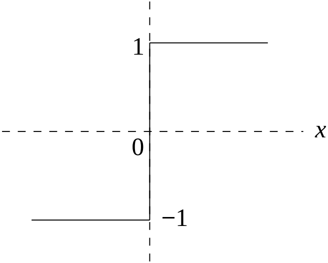
Example: Absolute value.

$|x|$ is not differentiable at $x = 0$ but subdifferentiable at $x = 0$:

$$|x| = \begin{cases} x & \text{if } x > 0, \\ 0 & \text{if } x = 0, \\ -x & \text{if } x < 0. \end{cases} \qquad \partial|x| = \begin{cases} \{+1\} & \text{if } x > 0, \\ [-1, +1] & \text{if } x = 0, \\ \{-1\} & \text{if } x < 0. \end{cases} \tag{9}$$



absolute value



subdifferential

Example: l_1-norm of an m-D vector $\mathbf{x}$.

$$\|\mathbf{x}\|_1 = \sum_{i=1}^{m} |x_i|, \quad \partial\|\mathbf{x}\|_1 = \sum_{i=1}^{m} \partial|x_i|. \tag{10}$$

Caution: Right-hand-side of Eq. 10 is **set addition**.
This gives a product of m sets, one for each element x_i:

$$\partial\|\mathbf{x}\|_1 = J_1 \times \cdots \times J_m, \quad J_i = \begin{cases} \{+1\} & \text{if } x_i > 0, \\ [-1, +1] & \text{if } x_i = 0, \\ \{-1\} & \text{if } x_i < 0. \end{cases} \tag{11}$$

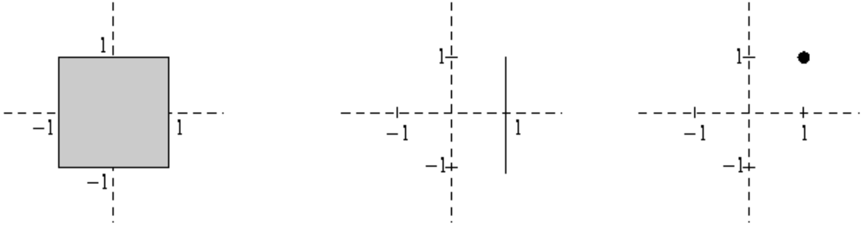
Alternatively, we can write $\partial\|\mathbf{x}\|_1$ as

$$\partial\|\mathbf{x}\|_1 = \{\, \mathbf{g} \,\} \text{ such that } g_i \begin{cases} = +1 & \text{if } x_i > 0, \\ \in [-1, +1] & \text{if } x_i = 0, \\ = -1 & \text{if } x_i < 0. \end{cases} \tag{12}$$

Another alternative:

$$\partial \|\mathbf{x}\|_1 = \{\, \mathbf{g} \,\} \text{ such that } \begin{cases} g_i = \operatorname{sgn}(g_i) & \text{if } |x_i| > 0, \\ |g_i| \le 1 & \text{if } x_i = 0. \end{cases} \tag{13}$$

Here are some examples of the subdifferentials of 2D l_1-norm.



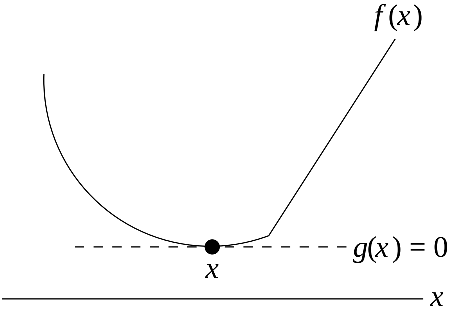
(a) $\partial f(0,0) = [-1,1] \times [-1,1]$

(b) $\partial f(1,0) = \{1\} \times [-1,1]$

(c) $\partial f(1,1) = \{(1,1)\}$

Optimality Condition

$\mathbf{x}$ is a minimum of f if $\mathbf{0} \in \partial f(\mathbf{x})$, i.e., $\mathbf{0}$ is a subgradient of f.

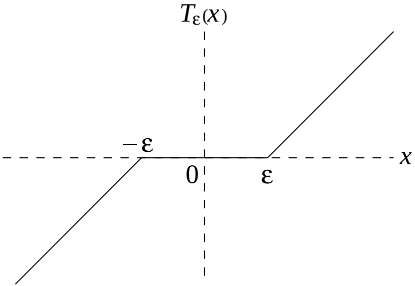


For more details on convex functions and convex optimization, refer to [Bertsekas2003, Boyd2004, Rockafellar70, Shor85].

Back to Robust PCA

Shrinkage or **soft threshold** operator:

$$T_\varepsilon(x) = \begin{cases} x - \varepsilon & \text{if } x > \varepsilon, \\ x + \varepsilon & \text{if } x < -\varepsilon, \\ 0 & \text{otherwise.} \end{cases} \tag{14}$$



Using convex optimization, the following minimizers are shown [Cai2010, Hale2007]:

For an $m \times n$ matrix $\mathbf{M}$ with SVD $\mathbf{U}\mathbf{S}\mathbf{V}^\top$,

$$\mathbf{U}T_\varepsilon(\mathbf{S})\mathbf{V}^\top = \arg\min_{\mathbf{X}} \varepsilon\|\mathbf{X}\|_* + \frac{1}{2}\|\mathbf{X} - \mathbf{M}\|_F^2, \tag{15}$$

$$T_\varepsilon(\mathbf{M}) = \arg\min_{\mathbf{X}} \varepsilon\|\mathbf{X}\|_1 + \frac{1}{2}\|\mathbf{X} - \mathbf{M}\|_F^2. \tag{16}$$

Solving Robust PCA

There are several ways to solve robust PCA (Problem 3) [Lin2009,Wright2009]:

- principal component pursuit
- iterative thresholding
- accelerated proximal gradient
- augmented Lagrange multipliers

Augmented Lagrange Multipliers

Consider the problem

$$\min_{\mathbf{x}} f(\mathbf{x}) \quad \text{subject to } c_j(\mathbf{x}) = 0, j = 1, \ldots, m. \tag{17}$$

This is a constrained optimization problem.

Lagrange multipliers method reformulates Problem 17 as:

$$\min_{\mathbf{x}} f(\mathbf{x}) + \sum_{j=1}^{m} \lambda_j c_j(\mathbf{x}) \tag{18}$$

with some constants λ_j.

Penalty method reformulates Problem 17 as:

$$\min_{\mathbf{x}} f(\mathbf{x}) + \mu \sum_{j=1}^{m} c_j^2(\mathbf{x}). \tag{19}$$

- Parameter μ increases over iteration, e.g., by factor of 10.
- Need $\mu \to \infty$ to get good solution.

Augmented Lagrange multipliers method combines Lagrange multipliers and penalty:

$$\min_{\mathbf{x}} f(\mathbf{x}) + \sum_{j=1}^{m} \lambda_j c_j(\mathbf{x}) + \frac{\mu}{2} \sum_{j=1}^{m} c_j^2(\mathbf{x}). \tag{20}$$

Denote $\boldsymbol{\lambda} = [\lambda_1 \ \cdots \ \lambda_m]^\top$, $\mathbf{c}(\mathbf{x}) = [c_1(\mathbf{x}) \ \cdots \ c_m(\mathbf{x})]^\top$.
Then, Problem 20 becomes

$$\min_{\mathbf{x}} f(\mathbf{x}) + \boldsymbol{\lambda}^\top \mathbf{c}(\mathbf{x}) + \frac{\mu}{2} \|\mathbf{c}(\mathbf{x})\|^2. \tag{21}$$

If the constraints form a matrix $\mathbf{C} = [c_{jk}]$, then define $\mathbf{\Lambda} = [\lambda_{jk}]$.

Then Problem 20 becomes

$$\min_{\mathbf{x}} f(\mathbf{x}) + \langle \mathbf{\Lambda}, \mathbf{C}(\mathbf{x}) \rangle + \frac{\mu}{2} \|\mathbf{C}(\mathbf{x})\|_F^2. \tag{22}$$

- $\langle \mathbf{\Lambda}, \mathbf{C} \rangle$ is sum of product of corresponding elements:

$$\langle \mathbf{\Lambda}, \mathbf{C} \rangle = \sum_j \sum_k \lambda_{jk} c_{jk}.$$

- $\|\mathbf{C}\|_F$ is the Frobenius norm:

$$\|\mathbf{C}\|_F^2 = \sum_j \sum_k c_{jk}^2.$$

Augmented Lagrange Multipliers (ALM) Method

1. Initialize $\mathbf{\Lambda}$, $\mu > 0$, $\rho \geq 1$.
2. Repeat until convergence:
   2.1 Compute $\mathbf{x} = \arg\min_{\mathbf{x}} L(\mathbf{x})$ where
   $$L(\mathbf{x}) = f(\mathbf{x}) + \langle \mathbf{\Lambda}, \mathbf{C}(\mathbf{x}) \rangle + \frac{\mu}{2} \|\mathbf{C}(\mathbf{x})\|_F^2.$$
   2.2 Update $\mathbf{\Lambda} = \mathbf{\Lambda} + \mu \mathbf{C}(\mathbf{x})$.
   2.3 Update $\mu = \rho\mu$.

What kind of optimization algorithm is this?

ALM does not need $\mu \to \infty$ to get good solution.
That means, can converge faster.

With ALM, robust PCA (Problem 3) is reformulated as

$$\min_{\mathbf{A},\mathbf{E}} \|\mathbf{A}\|_* + \lambda\|\mathbf{E}\|_1 + \langle \mathbf{Y}, \mathbf{D} - \mathbf{A} - \mathbf{E}\rangle + \frac{\mu}{2}\|\mathbf{D} - \mathbf{A} - \mathbf{E}\|_F^2, \tag{23}$$

- $\mathbf{Y}$ are the Lagrange multipliers.
- Constraints $\mathbf{C} = \mathbf{D} - \mathbf{A} - \mathbf{E}$.

To implement Step 2.1 of ALM, need to find minima for $\mathbf{A}$ and $\mathbf{E}$.
Adopt alternating optimization.

Trace of Matrix

The trace of a matrix $\mathbf{A}$ is the sum of its diagonal elements a_{ii}:

$$\mathrm{tr}(\mathbf{A}) = \sum_{i=1}^{n} a_{ii}. \tag{24}$$

Strictly speaking, scalar c and a 1×1 matrix $[\,c\,]$ are not the same thing. Nevertheless, since $\mathrm{tr}([\,c\,]) = c$, we often write, for simplicity $\mathrm{tr}(c) = c$.

Properties

- $\mathrm{tr}(\mathbf{A}) = \sum_{i=1}^{n} \lambda_i$, where λ_i are the eigenvalues of $\mathbf{A}$.
- $\mathrm{tr}(\mathbf{A} + \mathbf{B}) = \mathrm{tr}(\mathbf{A}) + \mathrm{tr}(\mathbf{B})$
- $\mathrm{tr}(c\mathbf{A}) = c\,\mathrm{tr}(\mathbf{A})$
- $\mathrm{tr}(\mathbf{A}) = \mathrm{tr}(\mathbf{A}^\top)$

- $\mathrm{tr}(\mathbf{ABCD}) = \mathrm{tr}(\mathbf{BCDA}) = \mathrm{tr}(\mathbf{CDAB}) = \mathrm{tr}(\mathbf{DABC})$
- $\mathrm{tr}(\mathbf{X}^\top\mathbf{Y}) = \mathrm{tr}(\mathbf{X}\mathbf{Y}^\top) = \mathrm{tr}(\mathbf{Y}^\top\mathbf{X}) = \mathrm{tr}(\mathbf{Y}\mathbf{X}^\top) = \displaystyle\sum_i \sum_j x_{ij} y_{ij}$

From Problem 23, the minimal $\mathbf{E}$ with other variables fixed is given by

$$
\begin{aligned}
& \min_{\mathbf{E}} \lambda\|\mathbf{E}\|_1 + \langle \mathbf{Y}, \mathbf{D} - \mathbf{A} - \mathbf{E}\rangle + \frac{\mu}{2}\|\mathbf{D} - \mathbf{A} - \mathbf{E}\|_F^2 \\
\Rightarrow \quad & \min_{\mathbf{E}} \lambda\|\mathbf{E}\|_1 + \mathrm{tr}(\mathbf{Y}^\top(\mathbf{D} - \mathbf{A} - \mathbf{E})) + \frac{\mu}{2}\|\mathbf{D} - \mathbf{A} - \mathbf{E}\|_F^2 \\
\Rightarrow \quad & \min_{\mathbf{E}} \lambda\|\mathbf{E}\|_1 + \mathrm{tr}(-\mathbf{Y}^\top\mathbf{E}) + \frac{\mu}{2}\mathrm{tr}((\mathbf{D} - \mathbf{A} - \mathbf{E})^\top(\mathbf{D} - \mathbf{A} - \mathbf{E})) \\
\vdots \quad & \text{(Homework)} \\
\Rightarrow \quad & \min_{\mathbf{E}} \lambda\|\mathbf{E}\|_1 + \frac{\mu}{2}\mathrm{tr}((\mathbf{E} - (\mathbf{D} - \mathbf{A} + \mathbf{Y}/\mu))^\top(\mathbf{E} - (\mathbf{D} - \mathbf{A} + \mathbf{Y}/\mu))) \\
\Rightarrow \quad & \min_{\mathbf{E}} \lambda\|\mathbf{E}\|_1 + \frac{\mu}{2}\|\mathbf{E} - (\mathbf{D} - \mathbf{A} + \mathbf{Y}/\mu)\|_F^2 \\
\Rightarrow \quad & \min_{\mathbf{E}} \frac{\lambda}{\mu}\|\mathbf{E}\|_1 + \frac{1}{2}\|\mathbf{E} - (\mathbf{D} - \mathbf{A} + \mathbf{Y}/\mu)\|_F^2. \qquad (25)
\end{aligned}
$$

Compare Eq. 16

$$T_\varepsilon(\mathbf{M}) = \arg\min_{\mathbf{X}} \varepsilon \|\mathbf{X}\|_1 + \frac{1}{2}\|\mathbf{X} - \mathbf{M}\|_F^2,$$

and Problem 25

$$\min_{\mathbf{E}} \frac{\lambda}{\mu}\|\mathbf{E}\|_1 + \frac{1}{2}\|\mathbf{E} - (\mathbf{D} - \mathbf{A} + \mathbf{Y}/\mu)\|_F^2.$$

Set $\mathbf{X} = \mathbf{E}$, $\mathbf{M} = \mathbf{D} - \mathbf{A} + \mathbf{Y}/\mu$. Then,

$$\mathbf{E} = T_{\lambda/\mu}(\mathbf{D} - \mathbf{A} + \mathbf{Y}/\mu). \tag{26}$$

From Problem 23, the minimal $\mathbf{A}$ with other variables fixed is given by

$$\begin{aligned}
& \min_{\mathbf{A}} \|\mathbf{A}\|_* + \langle \mathbf{Y}, \mathbf{D} - \mathbf{A} - \mathbf{E} \rangle + \frac{\mu}{2} \|\mathbf{D} - \mathbf{A} - \mathbf{E}\|_F^2 \\
\Rightarrow \quad & \min_{\mathbf{A}} + \operatorname{tr}(\mathbf{Y}^\top (\mathbf{D} - \mathbf{A} - \mathbf{E})) + \frac{\mu}{2} \|\mathbf{D} - \mathbf{A} - \mathbf{E}\|_F^2 \\
\vdots \quad & \text{(Homework)} \\
\Rightarrow \quad & \min_{\mathbf{A}} \frac{1}{\mu} \|\mathbf{A}\|_* + \frac{1}{2} \|\mathbf{A} - (\mathbf{D} - \mathbf{E} + \mathbf{Y}/\mu)\|_F^2 \qquad (27)
\end{aligned}$$

Compare Problem 27 and Eq. 15

$$\mathbf{U} T_\varepsilon(\mathbf{S}) \mathbf{V}^\top = \arg\min_{\mathbf{X}} \varepsilon \|\mathbf{X}\|_* + \frac{1}{2} \|\mathbf{X} - \mathbf{M}\|_F^2,$$

Set $\mathbf{X} = \mathbf{A}$, $\mathbf{M} = \mathbf{D} - \mathbf{E} + \mathbf{Y}/\mu$. Then,

$$\mathbf{A} = \mathbf{U}\, T_{1/\mu}(\mathbf{S})\, \mathbf{V}^\top. \qquad (28)$$

Robust PCA for Matrix Recovery via ALM

Inputs: $\mathbf{D}$.

1. Initialize $\mathbf{A} = \mathbf{0}$, $\mathbf{E} = \mathbf{0}$.
2. Initialize $\mathbf{Y}$, $\mu > 0$, $\rho > 1$.
3. Repeat until convergence:
4. Repeat until convergence:
5. $\mathbf{U}, \mathbf{S}, \mathbf{V} = \text{SVD}(\mathbf{D} - \mathbf{E} + \mathbf{Y}/\mu)$.
6. $\mathbf{A} = \mathbf{U}\, T_{1/\mu}(\mathbf{S})\, \mathbf{V}^\top$.
7. $\mathbf{E} = T_{\lambda/\mu}(\mathbf{D} - \mathbf{A} + \mathbf{Y}/\mu)$.
8. Update $\mathbf{Y} = \mathbf{Y} + \mu(\mathbf{D} - \mathbf{A} - \mathbf{E})$.
9. Update $\mu = \rho\mu$.

Outputs: $\mathbf{A}$, $\mathbf{E}$.

Typical initialization [Lin2009]:

- $\mathbf{Y} = \text{sgn}(\mathbf{D})/J(\text{sgn}(\mathbf{D}))$.
- $\text{sgn}(\mathbf{D})$ gives sign of each matrix element of $\mathbf{D}$.
- $J(\cdot)$ gives scaling factors:

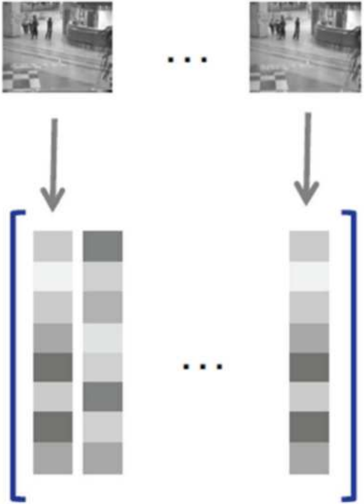
$$J(\mathbf{X}) = \max(\|\mathbf{X}\|_2, \lambda^{-1}\|\mathbf{X}\|_\infty).$$

- $\|\mathbf{X}\|_2$ is spectral norm, largest singular value of $\mathbf{X}$.
- $\|\mathbf{X}\|_\infty$ is largest absolute value of elements of $\mathbf{X}$.
- $\mu = 1.25\,\|\mathbf{D}\|_2$.
- $\rho = 1.5$.
- $\lambda = 1/\sqrt{\max(m, n)}$ for $m \times n$ matrix $\mathbf{D}$.

Example: Recovery of video background.

video frames

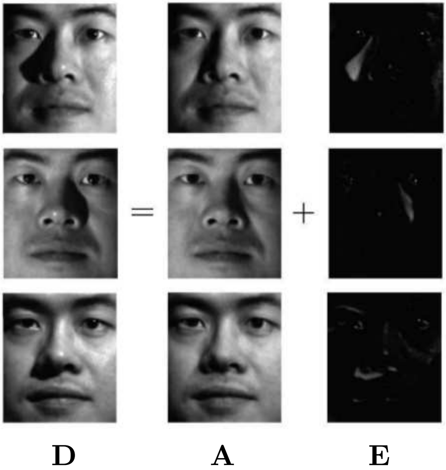
data matrix $\mathbf{D}$



Sample results:

data matrix | low-rank | error | low-rank | error

Example: Removal of specular reflection and shadow.



Fixed-Rank Robust PCA

In reflection removal, reflection may be global.

ground-truth

input

Then, $\mathbf{E}$ is not sparse: violate RPCA condition!
But, rank of $\mathbf{A} = 1$.

Fix the rank of $\mathbf{A}$ to deal with non-sparse $\mathbf{E}$ [Leow2013].

Fixed-Rank Robust PCA via ALM

Inputs: $\mathbf{D}$.

1. Initialize $\mathbf{A} = \mathbf{0}$, $\mathbf{E} = \mathbf{0}$.
2. Initialize $\mathbf{Y}$, $\mu > 0$, $\rho > 1$.
3. Repeat until convergence:
4. Repeat until convergence:
5. $\mathbf{U}, \mathbf{S}, \mathbf{V} = \mathrm{SVD}(\mathbf{D} - \mathbf{E} + \mathbf{Y}/\mu)$.
6. If $\mathrm{rank}(T_{1/\mu}(\mathbf{S})) < r$, $\mathbf{A} = \mathbf{U}\, T_{1/\mu}(\mathbf{S})\, \mathbf{V}^\top$; else $\mathbf{A} = \mathbf{U}\, \mathbf{S}_r \mathbf{V}^\top$.
7. $\mathbf{E} = T_{\lambda/\mu}(\mathbf{D} - \mathbf{A} + \mathbf{Y}/\mu)$.
8. Update $\mathbf{Y} = \mathbf{Y} + \mu(\mathbf{D} - \mathbf{A} - \mathbf{E})$.
9. Update $\mu = \rho\mu$.

Outputs: $\mathbf{A}$, $\mathbf{E}$.

$\mathbf{S}_r$ is $\mathbf{S}$ with last $m - r$ singular values set to 0.

Example: Removal of local reflections.

ground-truth

input

FRPCA

RPCA

Example: Removal of global reflections.

ground-truth input

FRPCA RPCA

Example: Removal of global reflections.

ground-truth

input

FRPCA

RPCA

Example: Background recovery for traffic video: fast moving vehicles.

input

FRPCA

RPCA

Example: Background recovery for traffic video: slow moving vehicles.

input

FRPCA

RPCA

Example: Background recovery for traffic video: temporary stop.

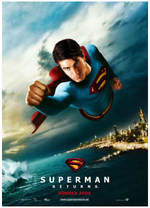
input

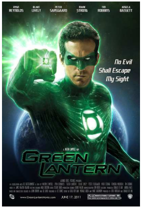
FRPCA

RPCA

Matrix Completion

Customers are asked to rate the movies from 1 (poor) to 5 (excellent).





A	5	5	3	2		
B		3	4			3
C	3			4	5	4
D	4		5	5	3	
⋮						

Customers rate only some movies ⇒ some data are missing.
How to estimate the missing data? matrix completion.

Let $\mathbf{D}$ denote data matrix with missing elements set to 0, and $M = \{(i, j)\}$ denote the indices of missing elements in $\mathbf{D}$.

Then, the **matrix completion** problem can be formulated as

Given $\mathbf{D}$ *and* M*, find matrix* $\mathbf{A}$ *that solves the problem*

$$\min_{\mathbf{A}} \|\mathbf{A}\|_* \quad \text{subject to } \mathbf{A} + \mathbf{E} = \mathbf{D}, E_{ij} = 0 \;\forall\, (i, j) \notin M. \tag{29}$$

- For $(i, j) \notin M$, constrain $E_{ij} = 0$ so that $A_{ij} = D_{ij}$; no change.
- For $(i, j) \in M$, $D_{ij} = 0$, i.e., $A_{ij} = E_{ij}$; recovered value.

Reformulating Problem 29 using augmented Lagrange multipliers gives

$$\min_{\mathbf{A}} \|\mathbf{A}\|_* + \langle \mathbf{Y}, \mathbf{D} - \mathbf{A} - \mathbf{E} \rangle + \frac{\mu}{2} \|\mathbf{D} - \mathbf{A} - \mathbf{E}\|_F^2 \tag{30}$$

such that $E_{ij} = 0 \;\forall\, (i, j) \notin M$.

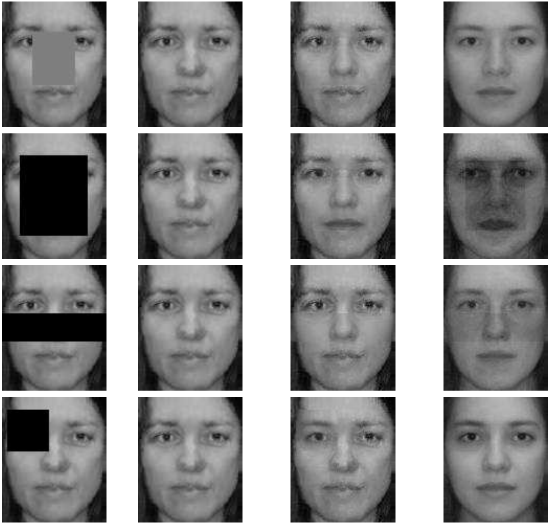
Robust PCA for Matrix Completion

Inputs: $\mathbf{D}$.

1. Initialize $\mathbf{A} = \mathbf{0}$, $\mathbf{E} = \mathbf{0}$.
2. Initialize $\mathbf{Y}$, $\mu > 0$, $\rho > 1$.
3. Repeat until convergence:
4. Repeat until convergence:
5. $\mathbf{U}, \mathbf{S}, \mathbf{V} = \mathrm{SVD}(\mathbf{D} - \mathbf{E} + \mathbf{Y}/\mu)$.
6. $\mathbf{A} = \mathbf{U}\, T_{1/\mu}(\mathbf{S})\, \mathbf{V}^\top$.
7. $\mathbf{E} = \Gamma_M(\mathbf{D} - \mathbf{A} + \mathbf{Y}/\mu)$, where
$$\Gamma_M(\mathbf{X}) = \begin{cases} X_{ij}, & \text{for } (i,j) \in M. \\ 0, & \text{for } (i,j) \notin M, \end{cases}$$
8. Update $\mathbf{Y} = \mathbf{Y} + \mu(\mathbf{D} - \mathbf{A} - \mathbf{E})$.
9. Update $\mu = \rho\mu$.

Outputs: $\mathbf{A}$, $\mathbf{E}$.

Example: Recovery of occluded parts in face images.



input | ground truth | matrix completion | matrix recovery

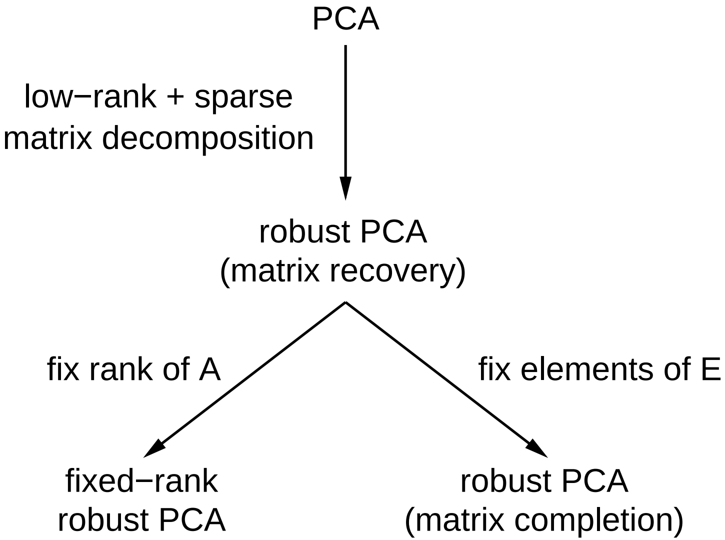
Summary

Robustification of PCA

PCA

- trimming → trimmed PCA
- 0–1 covariance → trimmed PCA
- weighted SVD → weighted PCA
- robust error function → winsorized PCA

Robust PCA


PCA
low−rank + sparse
matrix decomposition
robust PCA
(matrix recovery)
fix rank of A
fix elements of E
fixed−rank
robust PCA
robust PCA
(matrix completion)

Probing Questions

- If the data matrix of a problem is composed of a low-rank matrix, a sparse matrix and something else, can you still use robust PCA methods? If yes, how? If no, why?
- In application of robust PCA to high-resolution colour image processing, the data matrix contains three times as many rows as the number of pixels in the images, which can lead to a very large data matrix that takes a long time to compute. Suggest a way to overcome this problem.
- In application of robust PCA to video processing, the data matrix contains as many columns as the number of video frames, which can lead to a very large data matrix that is more than the available memory required to store the matrix. Suggest a way to overcome this problem.

Homework

1. Show that

$$\mathrm{tr}(\mathbf{X}^\top \mathbf{Y}) = \sum_i \sum_j x_{ij} y_{ij}.$$

2. Show that the following two optimization problems are equivalent:

$$\min_{\mathbf{E}} \lambda \|\mathbf{E}\|_1 + \mathrm{tr}(-\mathbf{Y}^\top \mathbf{E}) + \frac{\mu}{2} \mathrm{tr}((\mathbf{D} - \mathbf{A} - \mathbf{E})^\top (\mathbf{D} - \mathbf{A} - \mathbf{E}))$$

$$\min_{\mathbf{E}} \lambda \|\mathbf{E}\|_1 + \frac{\mu}{2} \mathrm{tr}((\mathbf{E} - (\mathbf{D} - \mathbf{A} + \mathbf{Y}/\mu))^\top (\mathbf{E} - (\mathbf{D} - \mathbf{A} + \mathbf{Y}/\mu)))$$

3. Show that the minimal $\mathbf{A}$ of Problem 23 with other variables fixed is given by

$$\min_{\mathbf{A}} \frac{1}{\mu} \|\mathbf{A}\|_* + \frac{1}{2} \|\mathbf{A} - (\mathbf{D} - \mathbf{E} + \mathbf{Y}/\mu)\|_F^2.$$

References I

1. D. P. Bertsekas. *Convex Analysis and Optimization.* Athena Scientific, 2003.
2. S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.
3. J. F. Cai, E. J. Candès, Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal of Optimization*, 20(4):1956–1982, 2010.
4. E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of ACM*, 58(3), 2011.
5. F. De la Torre and M. J. Black. Robust principal component analysis for computer vision. In *Proc. Int. Conf. Computer Vision*, 2001.
6. R. T. Rockafellar. *Convex Analysis.* Princeton University Press, 1970.

References II

7. K. Gabriel and S. Zamir. Lower rank approximation of matrices by least squares with any choice of weights. *Technometrics*, 21:489-498, 1979.

8. E. T. Hale, W. Yin, Y. Zhang. A fixed-point continuation method for l_1-regularized minimization with applications to compressed sensing. *Tech. Report TR07-07*, Dept. of Comp. and Applied Math., Rice University, 2007.

9. W. K. Leow, Y. Cheng, L. Zhang, T. Sim, and L. Foo. Background recovery by fixed-rank robust principal component analysis. In Proc. Int. Conf. on Computer Analysis of Images and Patterns, 2013.

10. Z. Lin, M. Chen, L. Wu, and Y. Ma. The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices. *Tech. Report UILU-ENG-09-2215*, UIUC, 2009.

11. N. Z. Shor. *Minimization Methods for Non-Differentiable Functions.* Springer-Verlag, 1985.

References III

12. J. Wright, Y. Peng, Y. Ma, A. Ganesh, and S. Rao. Robust principal component analysis: Exact recovery of corrupted low-rank matrices by convex optimization. In *Proc. Conf. on Neural Information Processing Systems*, 2009.

13. L. Xu and A. Yuille. Robust principal component analysis by self-organizing rules based on statistical physics approach. *IEEE Trans. Neural Networks*, 6(1):131-143, 1995.