

# Matrix Completion

CS 754 Lecture Notes

# Matrix Completion in Practice:

## Scenario 1

- Consider a survey of  $m$  people where each is asked  $q$  questions.
- It may not be possible to ask each person all  $q$  questions.
- Consider a matrix of size  $m$  by  $q$  (each row is the set of questions asked to any given person).
- This matrix is only partially filled (many missing entries).
- **Is it possible to infer the full matrix given just the recorded entries?**

# Matrix Completion in Practice:

## Scenario 2

- Some online shopping sites such as Amazon, Flipkart, Ebay, Netflix etc. have recommender systems.
- These websites collect product ratings from users (especially Netflix).
- Based on user ratings, these websites try to recommend other products/movies to the user that he/she will like with a high probability.
- Consider a matrix with the number of rows equal to the number of users, and number of columns equal to the number of movies/products.
- This matrix will be HIGHLY incomplete (no user has the patience to rate too many movies!!) – maybe only 5% of the entries will be filled up.
- **Can the recommender system infer user preferences from just the defined entries?**

# Matrix Completion in Practice:

## Scenario 2

- Read about the Netflix Prize to design a better recommender system:

[http://en.wikipedia.org/wiki/Netflix\\_Prize](http://en.wikipedia.org/wiki/Netflix_Prize)

# Matrix Completion in Practice:

## Scenario 3

- Consider an image or a video with several pixel values missing.
- This is not uncommon in range imagery or remote sensing applications!
- Consider a matrix whose each column is a (vectorized) patch of  $m$  pixels. Let the number of columns be  $K$ .
- This  $m$  by  $K$  matrix will have many missing entries.
- **Is it possible to infer the complete matrix given just the defined pixel values?**
- If the answer were yes, note the implications for image compression!

# Matrix Completion in Practice:

## Scenario 4

- Consider a long video sequence of  $F$  frames.
- Suppose I mark out  $M$  salient (interesting points)  $\{P_i\}$ ,  $1 \leq i \leq m$ , in the first frame.
- And track those points in all subsequent frames.
- Consider a matrix  $\mathbf{M}$  of size  $m \times 2F$  where row  $j$  contains the X and Y coordinates of points on the motion trajectory of initial point  $P_j$  (in each of the  $F$  frames).
- Unfortunately, many salient points may not be trackable due to occlusion or errors from the tracking algorithms.
- So  $\mathbf{M}$  is highly incomplete.
- **Is it possible to infer the true matrix from only the available measurements?**

# A property of these matrices

- Scenario 1: Many people will tend to give very similar or identical answers to many survey questions.
- Scenario 2: Many people will have similar preferences for movies (only a few factors affect user choices).
- Scenario 3: Non-local self-similarity!
- This makes the matrices in all these scenarios **(approximately) low in rank!**

# Scenario 5: Low rank matrices!

- Consider matrix **D** whose entry  $D_{ij}$  = euclidean distance between points  $\mathbf{p}_i$  and  $\mathbf{p}_j$  in some  $k$ -dimensional space.
- Such a matrix has rank at the most  $k+2$ .
- Proof:

$$D_{ij} = \|p_i - p_j\|^2 = \langle p_i, p_i \rangle + \langle p_j, p_j \rangle - 2\langle p_i, p_j \rangle$$

$$\mathbf{X} = (p_1 | p_2 | \dots | p_N), \mathbf{X} \in \mathbb{R}^{k \times N}$$

$$\therefore D = \text{diag}(\mathbf{X}\mathbf{X}^T)\mathbf{1}^T + \mathbf{1}\text{diag}(\mathbf{X}\mathbf{X}^T)^T - 2\mathbf{X}\mathbf{X}^T$$

$$\begin{aligned} \therefore \text{rank}(D) &\leq \text{rank}(\text{diag}(\mathbf{X}\mathbf{X}^T)\mathbf{1}^T) + \text{rank}(\mathbf{1}\text{diag}(\mathbf{X}\mathbf{X}^T)^T) + \text{rank}(-2\mathbf{X}\mathbf{X}^T) \\ &= 1 + 1 + k \end{aligned}$$

Column vector



# A property of these matrices

- Scenario 4: The true matrix underlying  $\mathbf{M}$  in question has been PROVED to be of **low rank** (in fact, **rank 3**) under orthographic projection (*ref: Tomasi and Kanade, "Shape and Motion from Image Streams Under Orthography: a Factorization Method", IJCV 1992*) and a few other more complex camera models (up to rank 9).
- In case of orthographic projection,  $\mathbf{M}$  can be expressed as a product of two matrices – a **rotation matrix** of size  $2F \times 3$ , and a **shape matrix** of size  $3 \times P$ . Hence it has rank 3.
- $\mathbf{M}$  is useful for many computer vision problems such as structure from motion, motion segmentation and multi-frame point correspondences.

# Low-rank matrices are cool!

- The answer to the four questions/scenarios is a NO in the general case.
- But it's a big YES if we assume that the underlying matrix has low rank (and which, as we have seen, is indeed the case for all four scenarios).
- Low rank matrices of size  $n_1 n_2$  with rank  $r$  have only  $(n_1 + n_2 - r)r$  degrees of freedom – much less than  $n_1 n_2$  when  $r$  is small.

# Theorem 1 (Informal Statement)

- Consider an unknown matrix  $\mathbf{M}$  of size  $n_1$  by  $n_2$  having rank  $r < \min(n_1, n_2)$ .
- Suppose we observe only a fraction of entries of  $\mathbf{M}$  in the form of matrix  $\mathbf{\Gamma}$ , where  $\mathbf{\Gamma}(i,j) = \mathbf{M}(i,j)$  for all  $(i,j)$  belonging to some set  $\Omega$  and  $\mathbf{\Gamma}(i,j)$  undefined elsewhere.
- If **(1)**  $\mathbf{M}$  has row and column spaces that are “sufficiently incoherent” with the canonical basis (i.e. identity matrix), **(2)**  $r$  is “sufficiently small”, and **(3)**  $\Omega$  is “sufficiently large”, then we can accurately recover  $\mathbf{M}$  from  $\mathbf{\Gamma}$  by solving the following rank minimization problem:

$$M^* = \min_M \text{rank}(M)$$

subject to

$$M(i,j) = \Gamma(i,j) \forall (i,j) \in \Omega$$

# Cool theorem, but ... ☹️

- The afore-mentioned optimization problem is NP-hard (in fact, current algorithms for solving it are known to have double exponential complexity in  $n_1$  or  $n_2$ !)

# Theorem 2 (Informal Statement)

- Consider an unknown matrix  $\mathbf{M}$  of size  $n_1$  by  $n_2$  having rank  $r < \min(n_1, n_2)$ .
- Suppose we observe only a fraction of entries of  $\mathbf{M}$  in the form of matrix  $\Gamma$ , where  $\Gamma(i,j) = \mathbf{M}(i,j)$  for all  $(i,j)$  belonging to some set  $\Omega$  and  $\Gamma(i,j)$  undefined elsewhere.
- If **(1)**  $\mathbf{M}$  has row and column spaces that are “sufficiently incoherent” with the canonical basis (i.e. identity matrix), **(2)**  $r$  is “sufficiently small”, and **(3)**  $\Omega$  is “sufficiently large”, then we can accurately recover  $\mathbf{M}$  from  $\Gamma$  by solving the following “**trace-norm**” minimization problem:

$$(Q0) : M^* = \min_M \|M\|_*$$

subject to

$$M(i,j) = \Gamma(i,j) \forall (i,j) \in \Omega$$

# What is the trace-norm of a matrix?

- The trace-norm of a matrix is the **sum of its singular values**.
- It is also called **nuclear norm**.
- It is a **softened version of the rank** of a matrix, just like the L1-norm of a vector is a softened version of the L0-norm of the vector.
- Minimization of the trace-norm (even under the given constraints) is a **convex optimization problem** and can be solved efficiently (no local minima issues).
- This is similar to the L1-norm optimization (in compressive sensing) being efficiently solvable.

# More about trace-norm minimization

- The efficient trace-norm minimization procedure is **provably** known to give the **EXACT SAME** result as the NP-hard rank minimization problem (under the same constraints and same conditions on the unknown matrix  $\mathbf{M}$  and the sampling set  $\Omega$ ).
- This is analogous to the case where L1-norm optimization yielded the same result as L0-norm optimization (under the same set of constraints and conditions).
- Henceforth we will concentrate only on Theorem 2 (and beyond).

# The devil is in the details

- **Beware:** Not all low-rank matrices can be recovered from partial measurements!
- Example consider a matrix containing zeroes everywhere except the top-right corner.
- This matrix is low rank, but it **cannot** be recovered from knowledge of only a fraction of its entries!
- Many other such examples exist.
- In reality, Theorems 1 and 2 work for low-rank matrices whose **singular vectors are sufficiently spread out**, i.e. **sufficiently incoherent with the canonical basis** (i.e. with the identity matrix).



# Coherence of a subspace

- The coherence of subspace  $\mathbf{U}$  of  $\mathbb{R}^n$  of dimension  $r$  with respect to the canonical basis  $\{\mathbf{e}_i\}$  is defined as:

$$\mu(U) = \frac{n}{r} \max_{1 \leq i \leq n} \|P_U \mathbf{e}_i\|^2$$

- Here  $P_U$  is the orthogonal projection onto  $\mathbf{U}$ . It is given by  $P_U = \mathbf{U}\mathbf{U}^T$ .

<http://www.math.lsa.umich.edu/~speyer/417/OrthoProj.pdf>

# Coherence of a basis

- We are interested in matrices whose row and column subspaces (i.e. the left and right singular vectors respectively) have low coherence w.r.t. canonical basis.
- Why? Because such matrices would not lie in the null-space of the sampling operator.

# Formal definition of key assumptions

- Consider an underlying matrix  $\mathbf{M}$  of size  $n_1$  by  $n_2$ . Let the SVD of  $\mathbf{M}$  be given as follows:

$$M = \sum_{k=1}^r \sigma_k u_k v_k^T = U \Sigma V^T, \Sigma = \text{diag}(\sigma_1, \sigma_2, \dots, \sigma_r, \dots, 0)$$

- We make the following assumptions about  $\mathbf{M}$ :

- (A0)**  $\exists \mu_0$  such that  $\max(\mu(U), \mu(V)) \leq \mu_0$
- (A1)** The maximum entry in the  $n_1$  by  $n_2$  matrix  $\sum_{k=1}^r u_k v_k^T$  is upper bounded by  $\mu_1 \sqrt{r/(n_1 n_2)}, \mu_1 > 0$

 In absolute value

# Theorem 2 (Formal Statement)

**Theorem 1.3** *Let  $\mathbf{M}$  be an  $n_1 \times n_2$  matrix of rank  $r$  obeying **A0** and **A1** and put  $n = \max(n_1, n_2)$ . Suppose we observe  $m$  entries of  $\mathbf{M}$  with locations sampled uniformly at random. Then there exist constants  $C, c$  such that if*

$$m \geq C \max(\mu_1^2, \mu_0^{1/2} \mu_1, \mu_0 n^{1/4}) nr(\beta \log n) \quad (1.9)$$

*for some  $\beta > 2$ , then the minimizer to the problem (1.5) is unique and equal to  $\mathbf{M}$  with probability at least  $1 - cn^{-\beta}$ . For  $r \leq \mu_0^{-1} n^{1/5}$  this estimate can be improved to*

$$m \geq C \mu_0 n^{6/5} r(\beta \log n) \quad (1.10)$$

*with the same probability of success.*

$$(Q0) : \mathbf{M}^* = \min_{\mathbf{M}} \|\mathbf{M}\|_*$$

subject to

$$\mathbf{M}(i,j) = \Gamma(i,j) \forall (i,j) \in \Omega$$

the trace-norm minimizer  
(in the informal  
statement of theorem 2)

Candes and Recht , *Exact matrix completion via convex optimization*,

<https://statweb.stanford.edu/~candes/papers/MatrixCompletion.pdf>

# Comments on Theorem 2

- Theorem 2 states that more entries of  $\mathbf{M}$  must be known (denoted by  $m$ ) for accurate reconstruction if (1)  $\mathbf{M}$  has larger rank  $r$ , (2) greater value of  $\mu_0$  in (A0), (3) greater value of  $\mu_1$  in (A1).
- Example: If  $\mu_0 = O(1)$  and the rank  $r$  is small, the reconstruction is accurate with high probability provided  $m \geq Cn^{1.2}r \log(n)$ .

# Comments on Theorem 2

- It turns out that if the singular vectors of matrix  $\mathbf{M}$  have bounded values, the condition (A1) almost always holds for the value  $\mu_1 = O(\log n)$ .
- The probabilistic nature of success of the recovery is because the entries in  $\Omega$  are from a uniform distribution. Recovery is successful for most such subsets of  $\Omega$ .

# Comments on Theorem 2

- Note that  $\Omega$  cannot afford to *completely* miss any row or any column of  $\mathbf{M}$  – otherwise recovery of even a rank-1 matrix is not possible.
- Thus one needs at least one entry from every row and every column – and hence one needs at least  $n \log n$  samples for this to happen.
- The latter is an example of the coupon collectors problem – see [https://en.wikipedia.org/wiki/Coupon\\_collector%27s\\_problem](https://en.wikipedia.org/wiki/Coupon_collector%27s_problem) . This shows that the bound on  $m$  is not too conservative.

# Another version of Theorem 2 = Theorem 2A

- Let  $\mathbf{M}$  be a  $n_1 \times n_2$  matrix of rank  $r$  obeying assumption (A2) below, of which only  $m$  entries are observed with locations chosen uniformly at random. Consider  $m \geq \beta C(\mu_B)^4 n \log^2 n$  where  $C$  is a constant and  $n = \max(n_1, n_2)$ . Then  $\mathbf{M}$  is the unique solution to Q0 with probability  $1 - 1/n^\beta$ .

Here assumption (A2) is that:

$$\exists \mu_B \text{ such that } \|u_k\|_\infty \leq \sqrt{\mu_B / n_1}, \|v_k\|_\infty \leq \sqrt{\mu_B / n_2}.$$

Candes and Plan, *Matrix completion under noise*, Proceedings of the IEEE



# Matrix Completion under noise

- Consider an *unknown* matrix  $\mathbf{M}$  of size  $n_1$  by  $n_2$  having rank  $r < \min(n_1, n_2)$ .
- Suppose we observe only a fraction of entries of  $\mathbf{M}$  in the form of matrix  $\mathbf{\Gamma}$ , where  $\mathbf{\Gamma}(i,j) = \mathbf{M}(i,j) + \mathbf{Z}(i,j)$  for all  $(i,j)$  belonging to some set  $\Omega$  and  $\mathbf{\Gamma}(i,j)$  undefined elsewhere.
- Here  $\mathbf{Z}$  refers to a white noise process which obeys the constraint that:

$$\sum_{(i,j) \in \Omega} Z_{ij}^2 \leq \delta, \delta > 0$$

# Matrix Completion under noise

- In such cases, the unknown matrix  $\mathbf{M}$  can be recovered by solving the following minimization procedure (called as a semi-definite program):

$$(Q1) : M^* = \min_M \|M\|_*$$

subject to

$$\sum_{(i,j) \in \Omega} (M(i,j) - \Gamma(i,j))^2 \leq \delta$$

# Theorem 3

- Under the assumptions for Theorems 1,2,2A, the reconstruction result from (Q1) is accurate with an error bound given by:

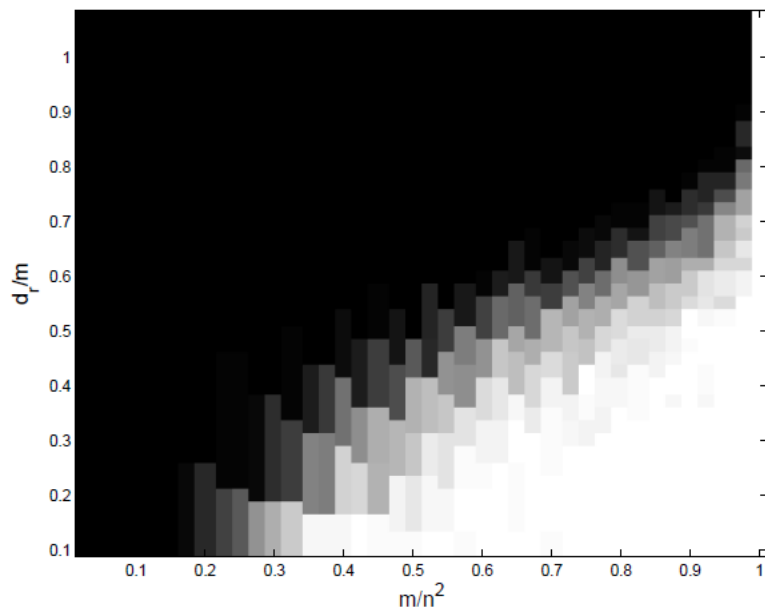
$$\|M - M^*\|_F^2 \leq 4 \sqrt{\frac{(2+p) \min(n_1, n_2)}{p}} \delta + 2\delta,$$

$$\text{where } p = \text{fraction of known entries} = \frac{m}{n_1 n_2} = \frac{|\Omega|}{n_1 n_2}$$

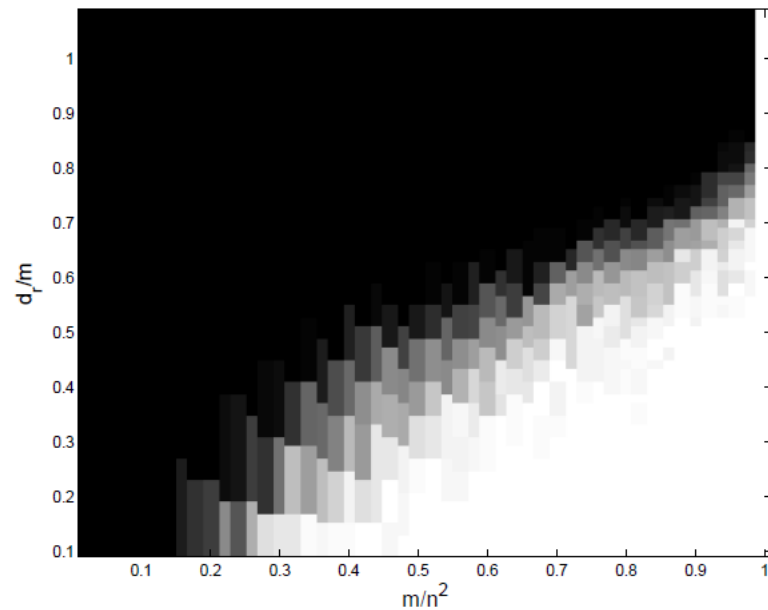
Candes and Plan, *Matrix completion under noise*, Proceedings of the IEEE

# Numerical results: Exact matrix completion

- Test data (low rank matrices of rank  $r$ ) generated synthetically using product of two random matrices of rank  $r$  each.
- Sample subset of entries uniformly at random.
- Estimation of underlying low-rank matrix using CVX.
- Study of variation of reconstruction error w.r.t.  $m$  and number of degrees of freedom ( $=r(2n-r)$  for a rank- $r$  matrix).
- Recorded: probability of success in recovery across 50 trials.



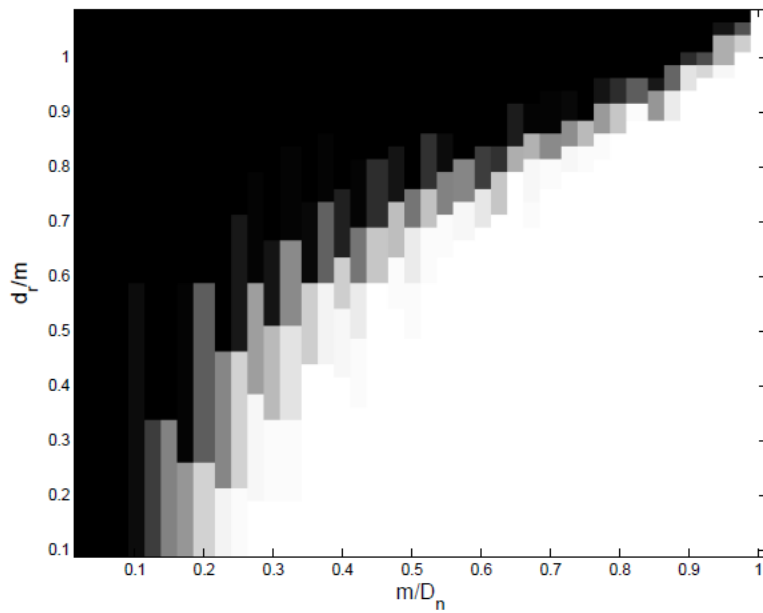
(a)



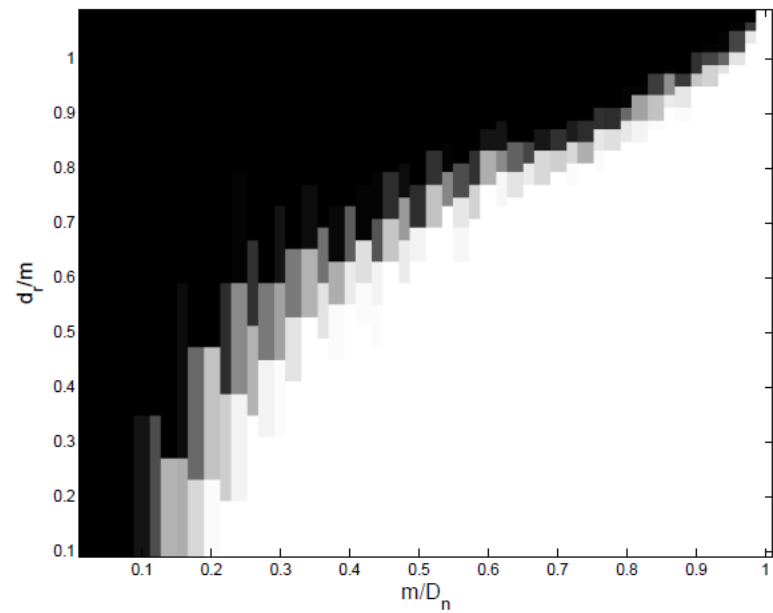
(b)

**Figure 1: Recovery of full matrices from their entries.** For each  $(n, m, r)$  triple, we repeated the following procedure 50 times. A matrix  $M$  of rank  $r$  and a subset of  $m$  entries were selected at random. Then we solved the nuclear norm minimization for  $X$  subject to  $X_{ij} = M_{ij}$  on the selected entries. We declared  $M$  to be recovered if  $\|X_{\text{opt}} - M\|_F / \|M\|_F < 10^{-3}$ . The results are shown for (a)  $n = 40$  and (b)  $n = 50$ . The color of each cell reflects the empirical recovery rate (scaled between 0 and 1). White denotes perfect recovery in all experiments, and black denotes failure for all experiments.

Candes and Recht , *Exact matrix completion via convex optimization*,  
<https://statweb.stanford.edu/~candes/papers/MatrixCompletion.pdf>



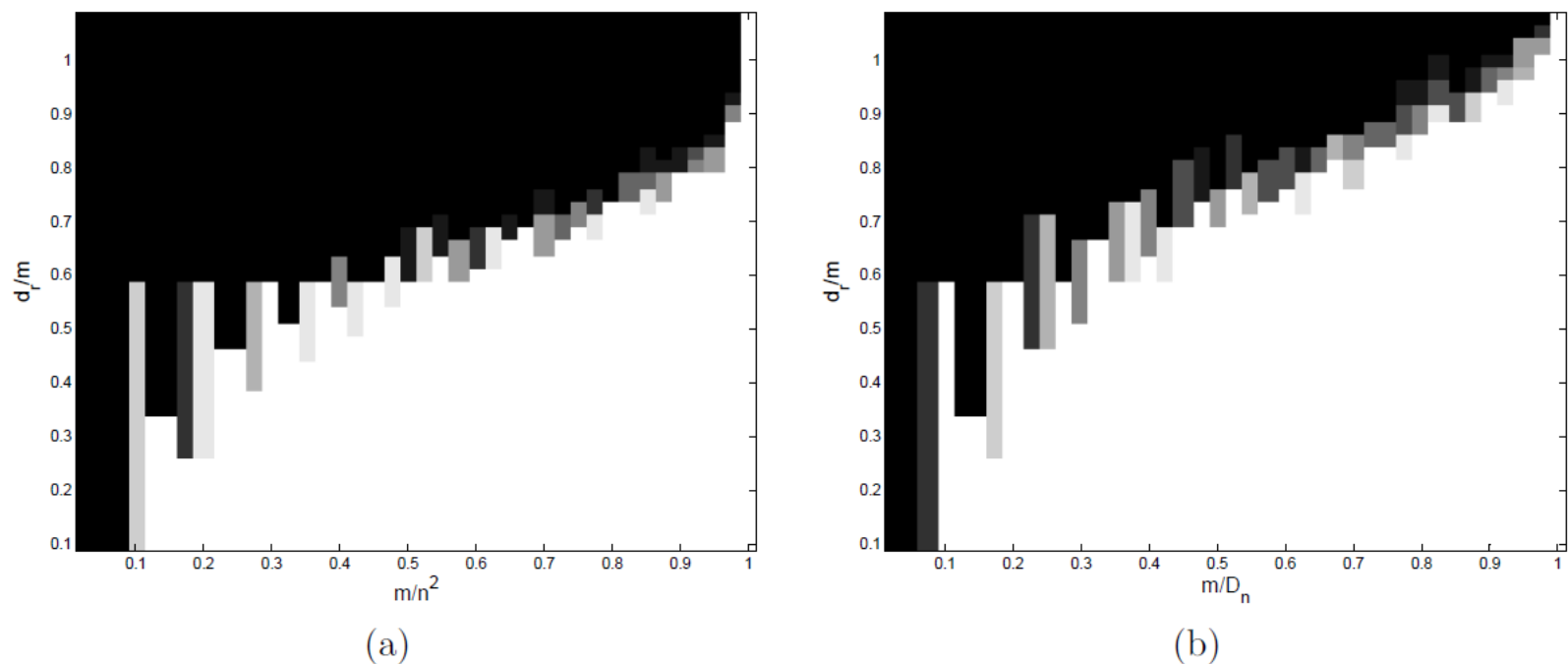
(a)



(b)

**Figure 2: Recovery of positive semidefinite matrices from their entries.** For each  $(n, m, r)$  triple, we repeated the following procedure 50 times. A positive semidefinite matrix  $M$  of rank  $r$  and a set of  $m$  entries were selected at random. Then we solved the nuclear norm minimization subject to  $X_{ij} = M_{ij}$  on the selected entries with the constraint that  $X \succeq 0$ . The color scheme for each cell denotes empirical recovery probability and is the same as in Figure 1. The results are shown for (a)  $n = 40$  and (b)  $n = 50$ .

Candes and Recht , *Exact matrix completion via convex optimization*,  
<https://statweb.stanford.edu/~candes/papers/MatrixCompletion.pdf>



**Figure 3: Recovery of matrices from Gaussian observations.** For each  $(n, m, r)$  triple, we repeated the following procedure 10 times. In (a), a matrix of rank  $r$  was generated as in Figures 1. In (b) a positive semidefinite matrix of rank  $r$  was generated as in Figures 2. In both plots, we select a matrix  $\mathcal{A}$  from the Gaussian ensemble with  $m$  rows and  $n^2$  (in (a)) or  $D_n = n(n+1)/2$  (in (b)) columns. Then we solve the nuclear norm minimization subject to  $\mathcal{A}(X) = \mathcal{A}(M)$ . The color scheme for each cell denotes empirical recovery probability and is the same as in Figures 1 and 2.

Candes and Recht , *Exact matrix completion via convex optimization*,  
<https://statweb.stanford.edu/~candes/papers/MatrixCompletion.pdf>

See the section on *(Compressive) Low Rank Matrix Recovery* later on in these slides.

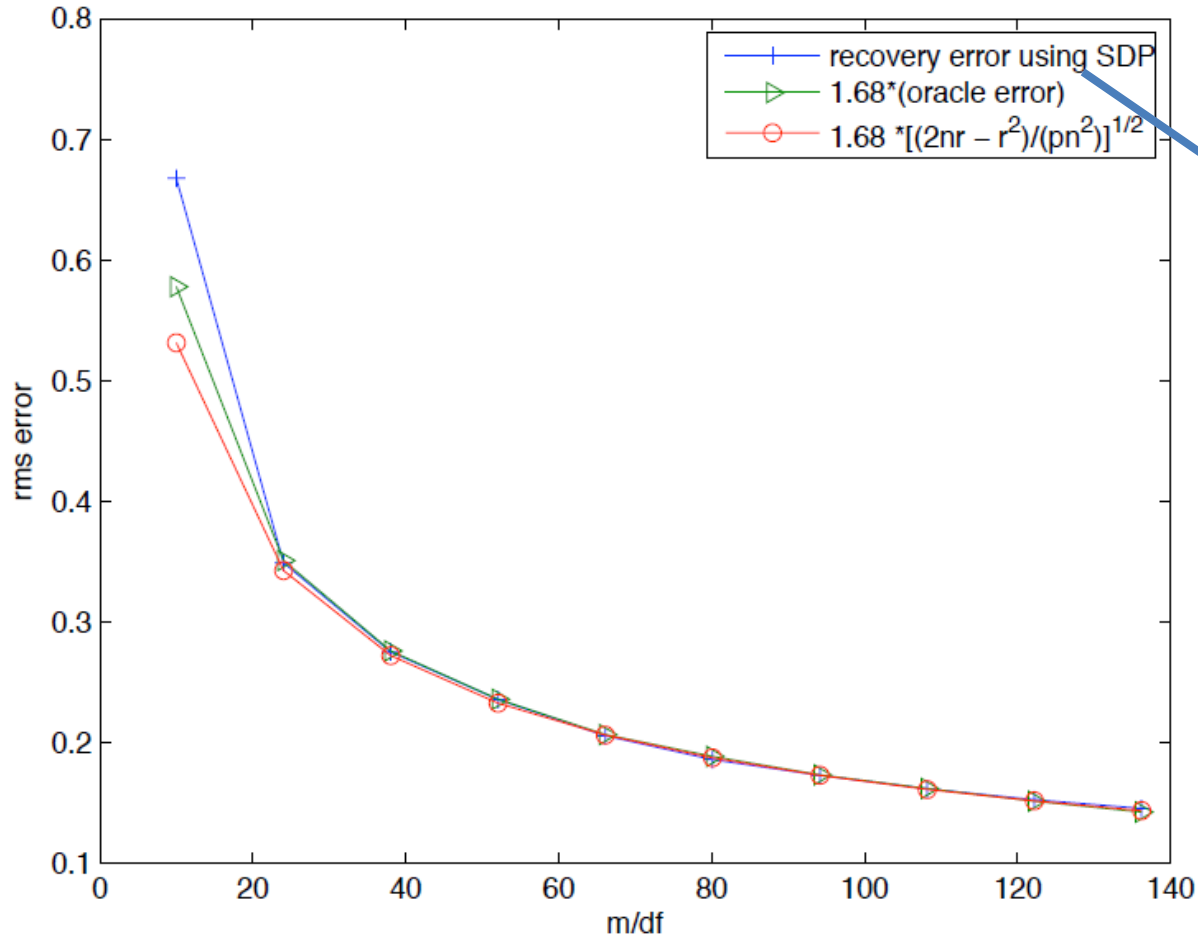
# Numerical results: noisy matrix completion

- Test matrices created by product of two Gaussian random matrices of rank  $r$ .
- Sampling set  $\Omega$  picked uniformly at random.
- Observations corrupted by Gaussian noise with mean 0 and  $\sigma = 1$ .
- Low rank matrix reconstructed as solution to:

$$J(M) = \frac{1}{2} \|P_{\Omega}(M - \Gamma)\|_F^2 + \mu \|M\|_*,$$
$$\mu = \sigma(\sqrt{n_1} + \sqrt{n_2})\sqrt{m} / \sqrt{n_1 n_2}$$



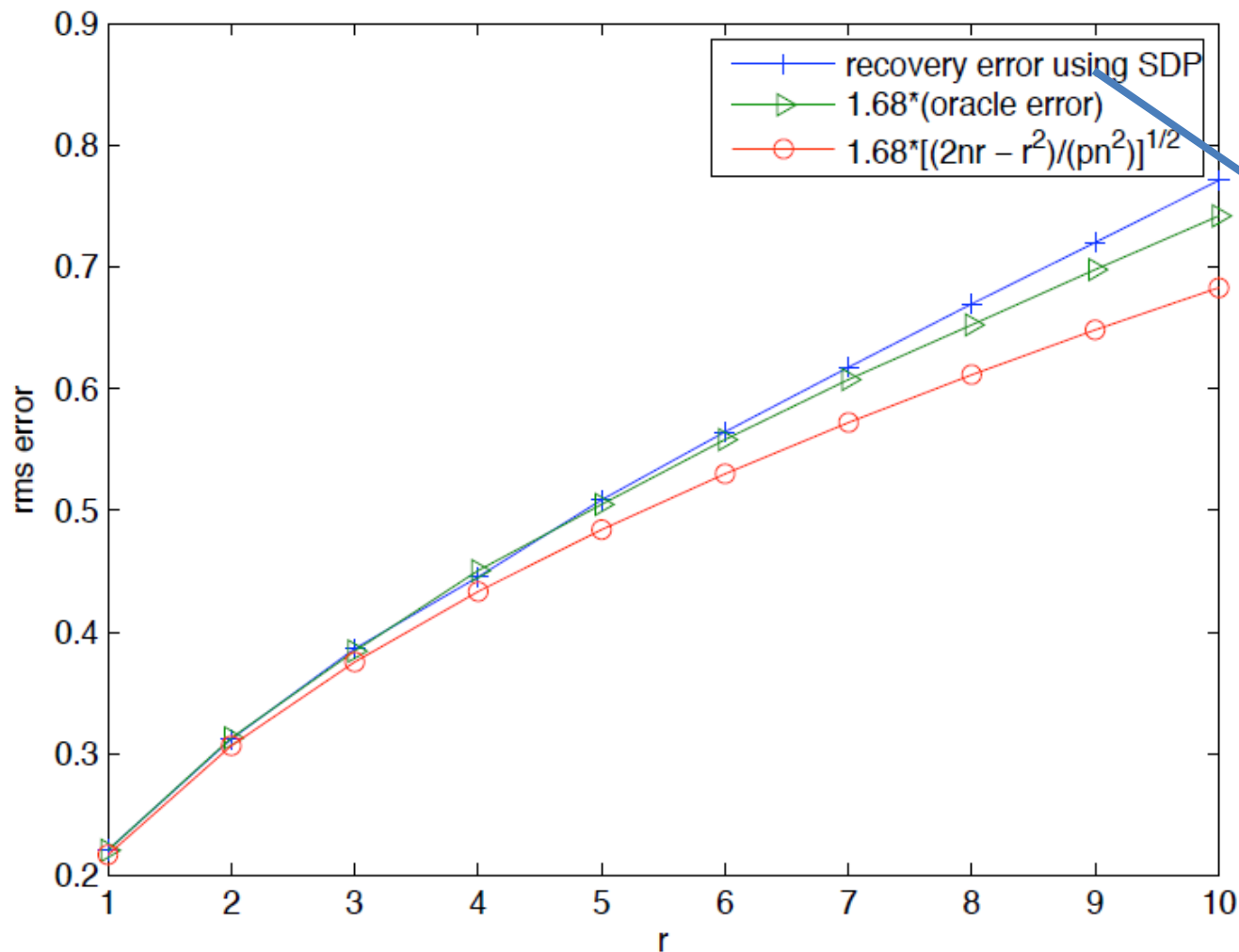
# Numerical results: noisy matrix completion



$$J(M) = \frac{1}{2} \|P_{\Omega}(M - \Gamma)\|_F^2 + \mu \|M\|_*,$$

$$\mu = \sigma(\sqrt{n_1} + \sqrt{n_2})\sqrt{m} / \sqrt{n_1 n_2}$$

# Numerical results: noisy matrix completion



$$J(M) = \frac{1}{2} \|P_{\Omega}(M - \Gamma)\|_F^2 + \mu \|M\|_*,$$
$$\mu = \sigma(\sqrt{n_1} + \sqrt{n_2})\sqrt{m} / \sqrt{n_1 n_2}$$

# A Minimization Algorithm

- Consider the minimization problem:

$$M^* = \min_M \|M\|_*$$

subject to

$$\sum_{(i,j) \in \Omega} (M(i,j) - \Gamma(i,j))^2 \leq \delta$$

- There are many techniques to solve this problem ([http://perception.csl.illinois.edu/matrix-rank/sample\\_code.html](http://perception.csl.illinois.edu/matrix-rank/sample_code.html))
- Out of these, we will study one method called “singular value thresholding”.

# Singular Value Thresholding (SVT)

```
 $\Phi^* = SVT(\Gamma, \tau > 0)$   
{  
   $Y^{(0)} = 0 \in R^{n_1 \times n_2}$   
   $k = 1$   
  while(convergence criterion not met)  
  {  
     $\Phi^{(k)} = \text{soft-threshold}(Y^{(k-1)}; \tau)$   
     $Y^{(k)} = Y^{(k-1)} + \delta_k P_{\Omega}(\Gamma - \Phi^{(k)}); k = k + 1;$   
  }  
   $\Phi^* = \Phi^{(k)};$   
}
```

```
soft-threshold ( $Y \in R^{n_1 \times n_2}; \tau$ )  
{  
   $Y = USV^T$  (using svd)  
  for ( $k = 1 : \text{rank}(Y)$ )  
     $S(k, k) = \max(0, S(k, k) - \tau);$   
}
```



The soft-thresholding procedure obeys the following property (which we state w/o proof).

$$\text{soft-threshold}(Y; \tau) = \arg \min_X \frac{1}{2} \|X - Y\|_F^2 + \tau \|X\|_*$$

# Properties of SVT (stated w/o proof)

- The sequence  $\{\Phi_k\}$  converges to the true solution of the problem below provided the step-sizes  $\{\delta_k\}$  all lie between 0 and 2.

$$M^* = \min_M \tau \|M\|_* + 0.5 \|M\|_F^2$$

subject to

$$\forall (i, j) \in \Omega, M(i, j) = \Gamma(i, j)$$

- For large values of  $\tau$ , this converges to the solution of the original problem (i.e. without the Frobenius norm term).

# Properties of SVT (stated w/o proof)

- The matrices  $\{\Phi_k\}$  turn out to have low rank (empirical observation – proof not established).
- The matrices  $\{Y_k\}$  also turn out to be sparse (empirical observation – rigorous proof not established).
- The SVT step does not require computation of full SVD – we need only those singular vectors whose singular values exceed  $\tau$ . There are special iterative methods for that.

# Results

- The SVT algorithm works very efficiently and is easily implementable in MATLAB.
- The authors report reconstruction of a 30,000 by 30,000 matrix in just 17 minutes on a 1.86 GHz dual-core desktop with 3 GB RAM and with MATLAB's multithreading option enabled.

# Results (Data without noise)

Unknown $\mathbf{M}$				Computational results		
size ( $n \times n$ )	rank ( $r$ )	$m/d_r$	$m/n^2$	time(s)	# iters	relative error
1,000 $\times$ 1,000	10	6	0.12	23	117	$1.64 \times 10^{-4}$
	50	4	0.39	196	114	$1.59 \times 10^{-4}$
	100	3	0.57	501	129	$1.68 \times 10^{-4}$
5,000 $\times$ 5,000	10	6	0.024	147	123	$1.73 \times 10^{-4}$
	50	5	0.10	950	108	$1.61 \times 10^{-4}$
	100	4	0.158	3,339	123	$1.72 \times 10^{-4}$
10,000 $\times$ 10,000	10	6	0.012	281	123	$1.73 \times 10^{-4}$
	50	5	0.050	2,096	110	$1.65 \times 10^{-4}$
	100	4	0.080	7,059	127	$1.79 \times 10^{-4}$
20,000 $\times$ 20,000	10	6	0.006	588	124	$1.73 \times 10^{-4}$
	50	5	0.025	4,581	111	$1.66 \times 10^{-4}$
30,000 $\times$ 30,000	10	6	0.004	1,030	125	$1.73 \times 10^{-4}$

TABLE 5.1

*Experimental results for matrix completion. The rank  $r$  is the rank of the unknown matrix  $\mathbf{M}$ ,  $m/d_r$  is the ratio between the number of sampled entries and the number of degrees of freedom in an  $n \times n$  matrix of rank  $r$  (oversampling ratio), and  $m/n^2$  is the fraction of observed entries. All the computational results on the right are averaged over five runs.*

<https://arxiv.org/abs/0810.3286>



# Results (Noisy Data)

noise ratio	Unknown matrix $M$				Computational results		
	size ( $n \times n$ )	rank ( $r$ )	$m/d_r$	$m/n^2$	time(s)	# iters	relative error
$10^{-2}$	$1,000 \times 1,000$	10	6	0.12	10.8	51	$0.78 \times 10^{-2}$
		50	4	0.39	87.7	48	$0.95 \times 10^{-2}$
		100	3	0.57	216	50	$1.13 \times 10^{-2}$
$10^{-1}$	$1,000 \times 1,000$	10	6	0.12	4.0	19	$0.72 \times 10^{-1}$
		50	4	0.39	33.2	17	$0.89 \times 10^{-1}$
		100	3	0.57	85.2	17	$1.01 \times 10^{-1}$
1	$1,000 \times 1,000$	10	6	0.12	0.9	3	0.52
		50	4	0.39	7.8	3	0.63
		100	3	0.57	34.8	3	0.69

TABLE 5.3

*Simulation results for noisy data. The computational results are averaged over five runs. For each test, the table shows the results of Algorithm 1 applied with an early stopping criterion*

<https://arxiv.org/abs/0810.3286>

# Results on real data

- Dataset consists of a matrix **M** of geodesic distances between 312 cities in the USA/Canada.
- This matrix is of approximately low-rank (in fact, the relative Frobenius error between **M** and its rank-3 approximation is 0.1159).
- 70% of the entries of this matrix (chosen uniformly at random) were blanked out.

# Results on real data

- The underlying matrix was estimated using SVT.
- In just a few seconds and a few iterations, the SVT produces an estimate that is as accurate as the best rank-3 approximation of **M**.

# Results on real data

Algorithm	rank	$k_i$	time	$\ \mathbf{M} - \mathbf{M}_i\ _F / \ \mathbf{M}\ _F$	$\ \mathbf{M} - \mathbf{X}^{k_i}\ _F / \ \mathbf{M}\ _F$
SVT	1	58	1.4	0.4091	0.4170
	2	190	4.8	0.1895	0.1980
	3	343	8.9	0.1159	0.1252
(3.6)	1	47	2.6	0.4091	0.4234
	2	166	7.2	0.1895	0.1998
	3	310	13.3	0.1159	0.1270

TABLE 5.5

*Speed and accuracy of the completion of the city-to-city distance matrix. Here,  $\|\mathbf{M} - \mathbf{M}_i\|_F / \|\mathbf{M}\|_F$  is the best possible relative error achieved by a matrix of rank  $i$ .*

<https://arxiv.org/abs/0810.3286>

# Robust Principal Components Analysis

# Basic Question

- Consider a matrix **M** of size  $n_1 \times n_2$  that is the sum of two components – **L** (a low-rank components) and **S** (a component with sparse but unknown support).
- Can we recover **L** and **S** given only **M**?
- Note: support of **S** means all those matrix entries  $(i,j)$  where  $S(i,j) \neq 0$ .

# Why ask this question? Scenario 1

- Consider a video taken with a static camera (like at airports or on overhead bridges on expressways).
- Such videos contain a “background” portion that is static or changes infrequently, and a moving portion called “foreground” (people, cars etc.).
- In many applications (esp. surveillance for security or traffic monitoring), we need to detect and track the moving foreground.

# Why ask this question? Scenario 1

- Let us represent this video as a  $N \times T$  matrix, where the number of columns  $T$  equals the number of frames in the video, and where each column containing a single video-frame of  $N = N_1 \times N_2$  pixels converted to a vector form.
- This matrix can clearly be expressed as the sum of a low-rank background matrix and a sparse foreground matrix.



# Why ask this question? Scenario 2

- It is well-known that the images of a convex Lambertian object taken from the same viewpoint, but under  $K$  different lighting directions, span a low-dimensional space (ref: Basri and Jacobs, “Lambertian reflectance and linear subspaces”, IEEE TPAMI, 2003).
- This means that a matrix **M** of size  $N \times K$  (each column of which is an image captured under one of the  $K$  lighting directions and represented in vectorized form) has a low approximate rank – in fact at most rank 9.

# Why ask this question? Scenario 2

- But many objects of interest (such as human faces) are neither convex nor exactly Lambertian.
- Effects such as shadows, specularities or saturation cause violation of the low-rank assumption.
- But these errors have sparse spatial support, even though their magnitudes are large.
- Here again,  $M$  can be represented as a low-rank background and a sparse foreground.

# Theorem 1 (Informal Statement)

- Consider a matrix  $\mathbf{M}$  of size  $n_1$  by  $n_2$  which is the sum of a “sufficiently low-rank” component  $\mathbf{L}$  and a “sufficiently sparse” component  $\mathbf{S}$  whose support is uniformly randomly distributed in the entries of  $\mathbf{M}$ .
- Then the solution of the following optimization problem (known as principal component pursuit) yields **exact estimates** of  $\mathbf{L}$  and  $\mathbf{S}$  with “very high” probability:

$$E(L', S') = \min_{(L, S)} \|L\|_* + \frac{1}{\sqrt{\max(n_1, n_2)}} \|S\|_1$$

subject to  $L + S = M$ .

$$\text{Note : } \|S\|_1 = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} |S_{ij}|$$

This is a convex optimization problem.

# Be careful!

- There is an implicit assumption in here that the low-rank component is not sparse (e.g.: a matrix containing zeros everywhere except the right top corner won't do!).
- And that the sparse component is not low-rank (that's why we said that it's support is drawn from a uniform random distribution).

## Be careful!

- In addition, the low-rank component  $L$  must obey the assumptions  $A0$  and  $A1$  (required in matrix completion theory).

$$L_0 = U\Sigma V^* = \sum_{i=1}^r \sigma_i u_i v_i^*,$$

$$\max_i \|U^* e_i\|^2 \leq \frac{\mu r}{n_1}, \quad \max_i \|V^* e_i\|^2 \leq \frac{\mu r}{n_2}, \quad (1.2)$$

$$\|UV^*\|_\infty \leq \sqrt{\frac{\mu r}{n_1 n_2}}. \quad (1.3)$$

# Theorem 1 (Formal Statement)

**Theorem 1.1** *Suppose  $L_0$  is  $n \times n$ , obeys (1.2)–(1.3), and that the support set of  $S_0$  is uniformly distributed among all sets of cardinality  $m$ . Then there is a numerical constant  $c$  such that with probability at least  $1 - cn^{-10}$  (over the choice of support of  $S_0$ ), Principal Component Pursuit (1.1) with  $\lambda = 1/\sqrt{n}$  is exact, i.e.  $\hat{L} = L_0$  and  $\hat{S} = S_0$ , provided that*

$$\text{rank}(L_0) \leq \rho_r n \mu^{-1} (\log n)^{-2} \quad \text{and} \quad m \leq \rho_s n^2. \quad (1.4)$$

*Above,  $\rho_r$  and  $\rho_s$  are positive numerical constants. In the general rectangular case where  $L_0$  is  $n_1 \times n_2$ , PCP with  $\lambda = 1/\sqrt{n_{(1)}}$  succeeds with probability at least  $1 - cn_{(1)}^{-10}$ , provided that  $\text{rank}(L_0) \leq \rho_r n_{(2)} \mu^{-1} (\log n_{(1)})^{-2}$  and  $m \leq \rho_s n_1 n_2$ .*

$$n_{(2)} = \min(n_1, n_2) \quad n_{(1)} = \max(n_1, n_2)$$

Candes et al, *Robust Principal Component Analysis?*

<https://statweb.stanford.edu/~candes/papers/RobustPCA.pdf>

# Comments on Theorem 1

- If the value  $m$  is smaller, the upper bound on the maximum rank allowed is larger.
- We require the support of  $\mathbf{S}$  to be sparse and uniformly distributed but make no assumption about the magnitude/sign of the non-zero elements in  $\mathbf{S}$ . But we don't know the support of  $\mathbf{S}$ .
- In the matrix completion problem, we knew which entries in the matrix were “missing”. Here we do not know the support of  $\mathbf{S}$ . And the entries in  $\mathbf{S}$  are not “missing” but unknown and grossly corrupted!
- The probabilistic nature of the theorem is because of the support of  $\mathbf{S}$  is a uniformly randomly drawn subset of  $[1, 2, \dots, n_1] \times [1, 2, \dots, n_2]$ .

# Matrix Recovery with Corrupted as well as missing data

- In some cases, we may have missing entries in the matrix (with known location) in addition to corrupted entries (whose location and magnitude we DON'T know).
- In such cases, we solve:

$$E(L', S') = \min_{(L, S)} \|L\|_* + \frac{1}{\sqrt{\max(n_1, n_2)}} \|S\|_1$$

subject to  $P_\Omega(L + S) = M$ , where

$P_\Omega(X) = X_{ij}$  (where  $(i, j) \in \Omega$ ) and 0 elsewhere.



# Theorem 2 (Formal Statement)

**Theorem 1.2** *Suppose  $L_0$  is  $n \times n$ , obeys the conditions (1.2)–(1.3), and that  $\Omega_{obs}$  is uniformly distributed among all sets of cardinality  $m$  obeying  $m = 0.1n^2$ . Suppose for simplicity, that each observed entry is corrupted with probability  $\tau$  independently of the others. Then there is a numerical constant  $c$  such that with probability at least  $1 - cn^{-10}$ , Principal Component Pursuit (1.5) with  $\lambda = 1/\sqrt{0.1n}$  is exact, i.e.  $\hat{L} = L_0$ , provided that*

$$\text{rank}(L_0) \leq \rho_r n \mu^{-1} (\log n)^{-2}, \quad \text{and} \quad \tau \leq \tau_s. \quad (1.6)$$

*Above,  $\rho_r$  and  $\tau_s$  are positive numerical constants. For general  $n_1 \times n_2$  rectangular matrices, PCP with  $\lambda = 1/\sqrt{0.1n_{(1)}}$  succeeds from  $m = 0.1n_1n_2$  corrupted entries with probability at least  $1 - cn_{(1)}^{-10}$ , provided that  $\text{rank}(L_0) \leq \rho_r n_{(2)} \mu^{-1} (\log n_{(1)})^{-2}$ .*

Candes et al, Robust Principal Component Analysis?

<https://statweb.stanford.edu/~candes/papers/RobustPCA.pdf>

# Comments on Theorem 2

- If all entries were observed (i.e.  $m = n_1 n_2$ ), then this reduces to Theorem 1.
- If  $\tau = 0$ , this becomes a pure matrix completion problem (no sparse component), but which is solved using a slightly different objective function (question: what's that difference?).
- In the  $\tau = 0$  case, the minimization routine is guaranteed to give us a low-rank component as usual, but a “sparse” component that contains all zeros.
- In this theorem, observe that the guarantees hold only for  $\mathbf{L}$ , and the recovery of  $\mathbf{S}$  is not possible, as many entries of  $\mathbf{S}$  may not be observed.
- The parameter for balancing between the  $\mathbf{L}$  and  $\mathbf{S}$  terms is auto-tuned and part of the mathematical analysis (no external tuning is required for the theorem to hold).

# Numerical Results on RPCA

- Test matrices created by adding together low rank and sparse matrices.
- **L** matrices created from the product of low-rank Gaussian random matrices.
- **S** matrices created from uniform random sampling with entries marked 0,+1,-1 with probability  $1-\rho$ ,  $\rho/2$ ,  $\rho/2$ .
- Success declared if relative error of low rank part is less than 0.001.
- A specific PCP algorithm is used.

# Numerical Results on RPCA

- In a second experiment, the sign of the sampled entries in  $\mathbf{S}$  to be equal to the sign of the corresponding entries in  $\mathbf{L}$ .
- In a third experiment, the  $\mathbf{S}$  part is 0 and only  $\mathbf{L}$  is estimated from its sampled version (i.e. the low rank matrix completion problem).

# Results

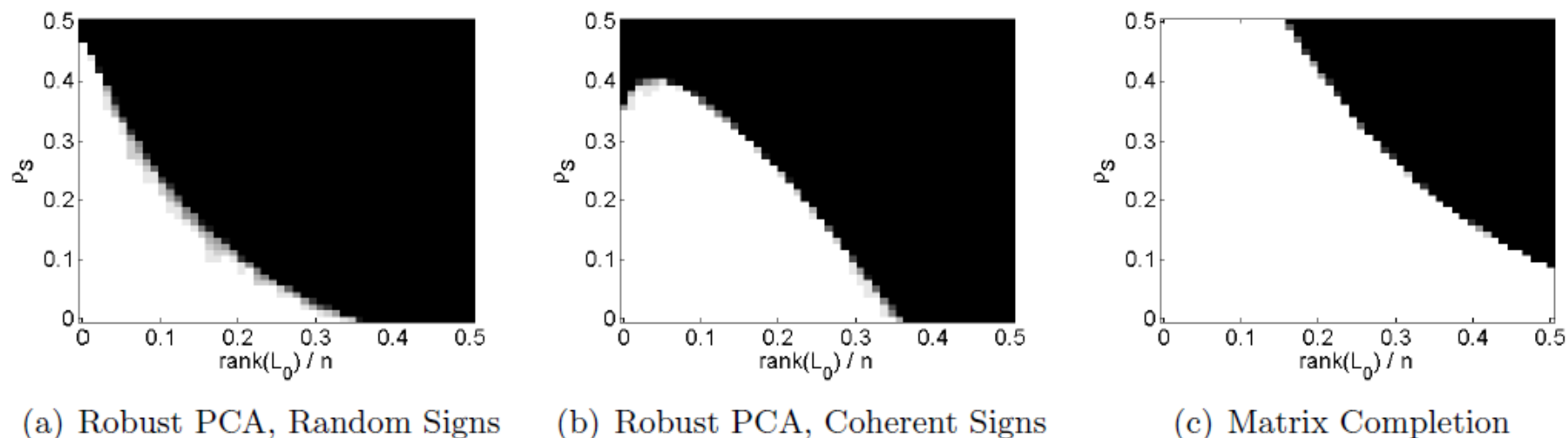
Dimension $n$	$\text{rank}(L_0)$	$\ S_0\ _0$	$\text{rank}(\hat{L})$	$\ \hat{S}\ _0$	$\frac{\ \hat{L}-L_0\ _F}{\ L_0\ _F}$	# SVD	Time(s)
500	25	12,500	25	12,500	$1.1 \times 10^{-6}$	16	2.9
1,000	50	50,000	50	50,000	$1.2 \times 10^{-6}$	16	12.4
2,000	100	200,000	100	200,000	$1.2 \times 10^{-6}$	16	61.8
3,000	250	450,000	250	450,000	$2.3 \times 10^{-6}$	15	185.2

$$\text{rank}(L_0) = 0.05 \times n, \|S_0\|_0 = 0.05 \times n^2.$$

Dimension $n$	$\text{rank}(L_0)$	$\ S_0\ _0$	$\text{rank}(\hat{L})$	$\ \hat{S}\ _0$	$\frac{\ \hat{L}-L_0\ _F}{\ L_0\ _F}$	# SVD	Time(s)
500	25	25,000	25	25,000	$1.2 \times 10^{-6}$	17	4.0
1,000	50	100,000	50	100,000	$2.4 \times 10^{-6}$	16	13.7
2,000	100	400,000	100	400,000	$2.4 \times 10^{-6}$	16	64.5
3,000	150	900,000	150	900,000	$2.5 \times 10^{-6}$	16	191.0

$$\text{rank}(L_0) = 0.05 \times n, \|S_0\|_0 = 0.10 \times n^2.$$

**Table 1:** Correct recovery for random problems of varying size. Here,  $L_0 = XY^* \in \mathbb{R}^{n \times n}$  with  $X, Y \in \mathbb{R}^{n \times r}$ ;  $X, Y$  have entries i.i.d.  $\mathcal{N}(0, 1/n)$ .  $S_0 \in \{-1, 0, 1\}^{n \times n}$  has support chosen uniformly at random and independent random signs;  $\|S_0\|_0$  is the number of nonzero entries in  $S_0$ . Top: recovering matrices of rank  $0.05 \times n$  from 5% gross errors. Bottom: recovering matrices of rank  $0.05 \times n$  from 10% gross errors. In all cases, the rank of  $L_0$  and  $\ell_0$ -norm of  $S_0$  are correctly estimated. Moreover, the number of partial singular value decompositions (# SVD) required to solve PCP is almost constant.



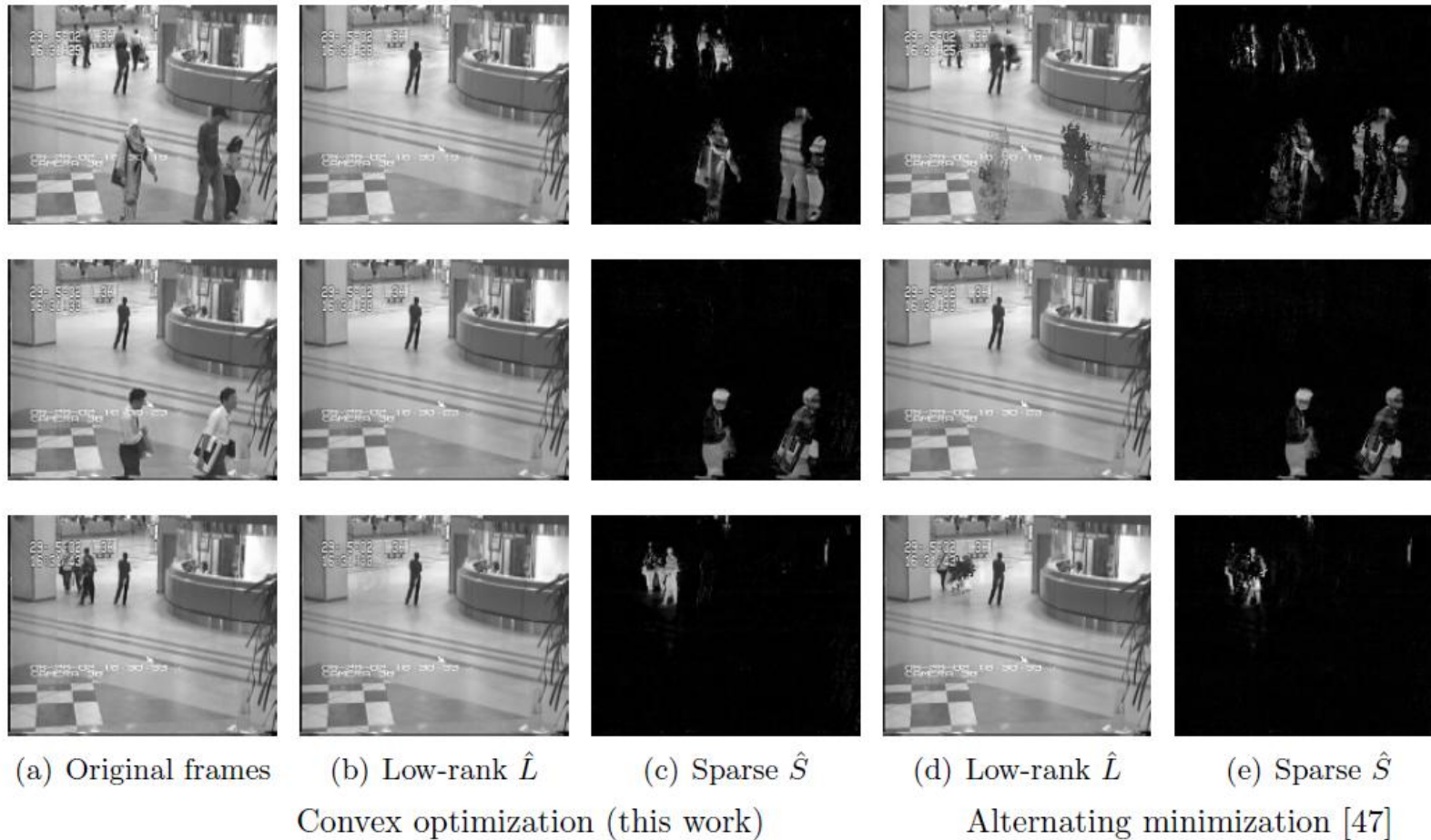
**Figure 1: Correct recovery for varying rank and sparsity.** Fraction of correct recoveries across 10 trials, as a function of  $\text{rank}(L_0)$  (x-axis) and sparsity of  $S_0$  (y-axis). Here,  $n_1 = n_2 = 400$ . In all cases,  $L_0 = XY^*$  is a product of independent  $n \times r$  i.i.d.  $\mathcal{N}(0, 1/n)$  matrices. Trials are considered successful if  $\|\hat{L} - L_0\|_F / \|L_0\|_F < 10^{-3}$ . Left: low-rank and sparse decomposition,  $\text{sgn}(S_0)$  random. Middle: low-rank and sparse decomposition,  $S_0 = \mathcal{P}_\Omega \text{sgn}(L_0)$ . Right: matrix completion. For matrix completion,  $\rho_s$  is the probability that an entry is omitted from the observation.

<https://statweb.stanford.edu/~candes/papers/RobustPCA.pdf>

# Results on background subtraction

- Experiments on video sequences acquired in a shopping mall.
- Video size is  $176 \times 144 \times 200$ , equal to a matrix of size  $25344 \times 200$ .
- Illumination change was minimal.
- Another experiment performed on a video of size  $168 \times 120 \times 250$  (matrix size  $20,160 \times 250$ ), with severe illumination changes.

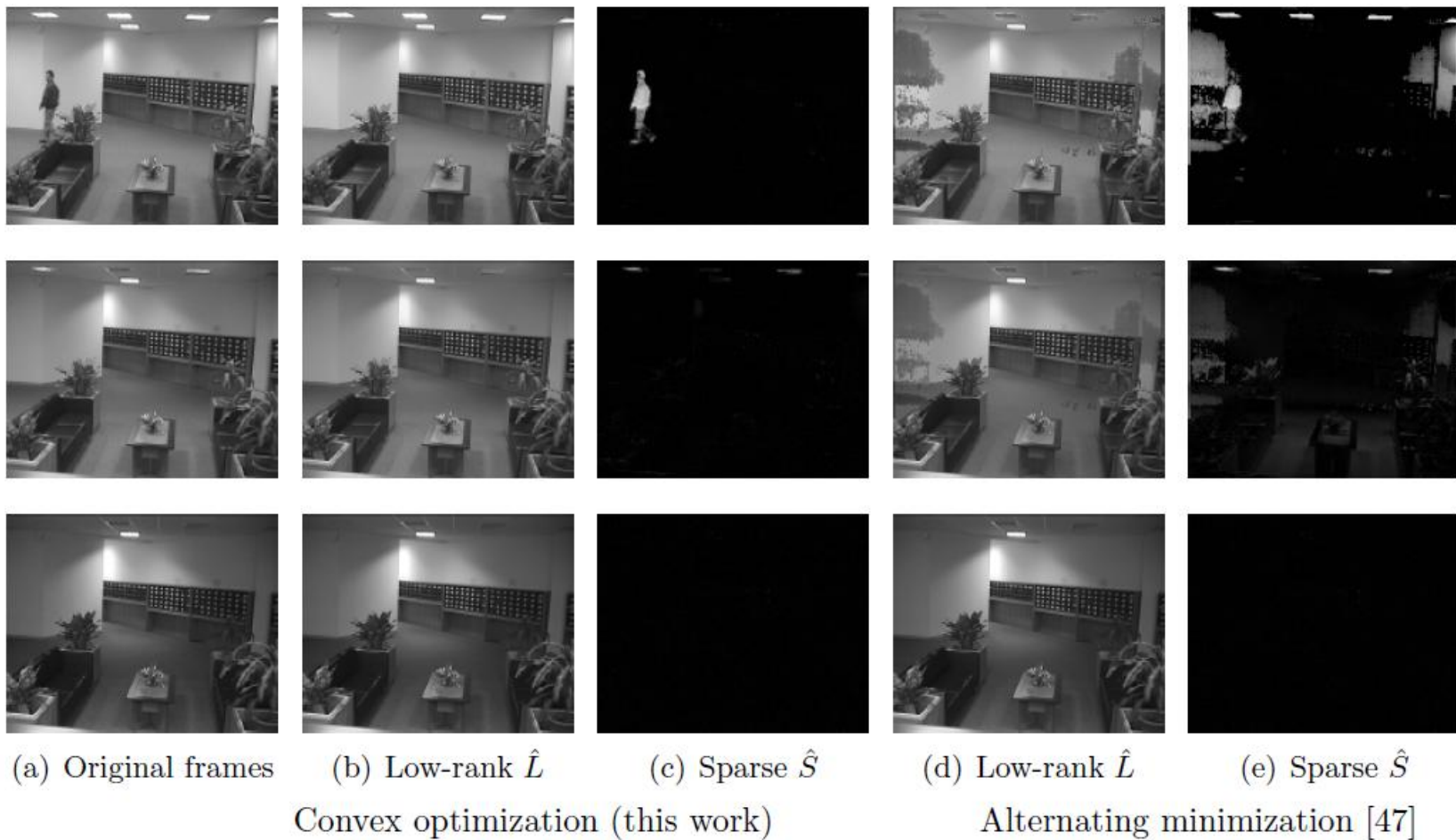
# Results on background subtraction



**Figure 2:** Background modeling from video. Three frames from a 200 frame video sequence taken in an airport [32]. (a) Frames of original video  $M$ . (b)-(c) Low-rank  $\hat{L}$  and sparse components  $\hat{S}$  obtained by PCP, (d)-(e) competing approach based on alternating minimization of an  $m$ -estimator [47]. PCP yields a much more appealing result despite using less prior knowledge.

<https://statweb.stanford.edu/~candes/papers/RobustPCA.pdf>

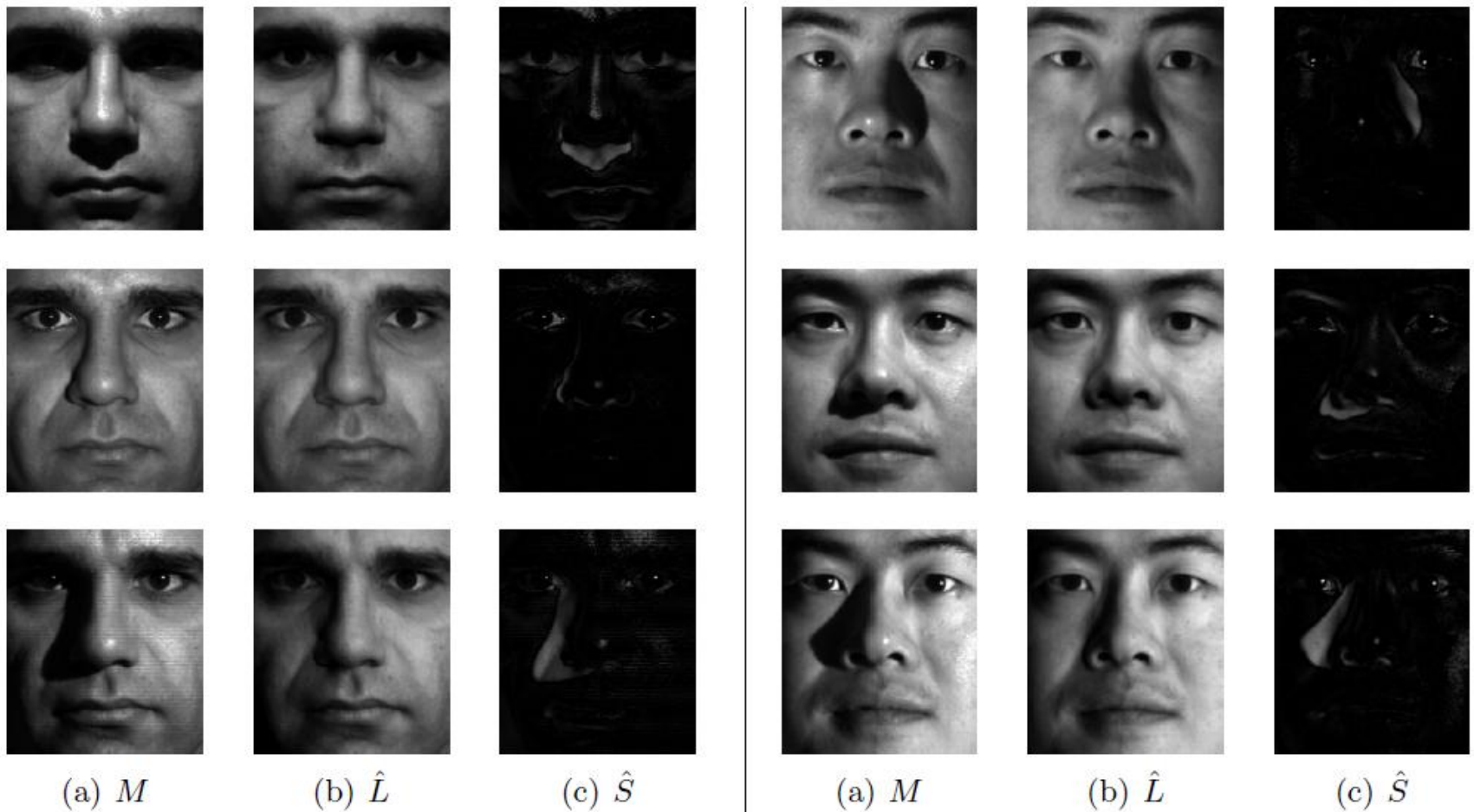




**Figure 3:** Background modeling from video. Three frames from a 250 frame sequence taken in a lobby, with varying illumination [32]. (a) Original video  $M$ . (b)-(c) Low-rank  $\hat{L}$  and sparse  $\hat{S}$  obtained by PCP. (d)-(e) Low-rank and sparse components obtained by a competing approach based on alternating minimization of an m-estimator [47]. Again, convex programming yields a more appealing result despite using less prior information.

# Results on shadow and specularities removal

- Experiments performed on images from the Yale face database.
- Image size is 192 x 168, images acquired under 58 different lighting conditions.
- Thus decomposition performed on a matrix of size 32,356 x 58.



**Figure 4:** Removing shadows, specularities, and saturations from face images. (a) Cropped and aligned images of a person's face under different illuminations from the Extended Yale B database. The size of each image is  $192 \times 168$  pixels, a total of 58 different illuminations were used for each person. (b) Low-rank approximation  $\hat{L}$  recovered by convex programming. (c) Sparse error  $\hat{S}$  corresponding to specularities in the eyes, shadows around the nose region, or brightness saturations on the face. Notice in the bottom left that the sparse term also compensates for errors in image acquisition.

# Algorithm for Robust PCA

- The algorithm uses the augmented Lagrangian technique.
- See [https://en.wikipedia.org/wiki/Augmented\\_Lagrangian\\_method](https://en.wikipedia.org/wiki/Augmented_Lagrangian_method) and [https://www.him.uni-bonn.de/fileadmin/him/Section6\\_HIM\\_v1.pdf](https://www.him.uni-bonn.de/fileadmin/him/Section6_HIM_v1.pdf)
- Suppose you want to solve:

$$\begin{array}{ll} \min & f(x) \\ \text{s.t.} & \forall i \in I, c_i(x) = 0 \end{array}$$

# Algorithm for Robust PCA

- Suppose you want to solve:

$$\min f(x)$$

$$\text{s.t. } \forall i \in I, c_i(x) = 0$$

- The augmented Lagrangian method (ALM) adopts the following iterative updates:

$$x_k = \arg \min f(x) + \underbrace{\mu_k \sum_{i \in I} c_i^2(x)}_{\text{Augmentation term}} - \underbrace{\sum_{i \in I} \lambda_i c_i(x)}_{\text{Lagrangian term}}$$

$$\lambda_i = \lambda_i - \mu_k c_i(x_k)$$

Augmentation term

Lagrangian term

# ALM: Some intuition

- What is the intuition behind the update of the Lagrange parameters  $\{\lambda_i\}$ ?
- The problem is:

$$\begin{aligned} \min f(x) \\ \text{s.t. } \forall i \in I, c_i(x) = 0 \end{aligned}$$

=

$$\begin{aligned} \min_x \max_{\lambda} f(x) + \lambda^t \mathbf{c}(x) \\ \mathbf{c}(x) = (c_1(x), c_2(x), \dots, c_{|I|}(x)) \end{aligned}$$

The maximum w.r.t.  $\lambda$  will be  $\infty$  unless the constraint is satisfied. Hence these problems are equivalent.


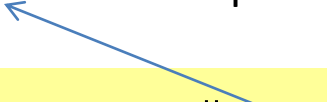
# ALM: Some intuition

- The problem is:

$$\begin{array}{ll} \min_x f(x) \\ \text{s.t. } \forall i \in I, c_i(x) = 0 \end{array} = \min_x \max_{\lambda} f(x) + \lambda^t \mathbf{c}(x)$$

$\mathbf{c}(x) = (c_1(x), c_2(x), \dots, c_{|I|}(x))$

Due to non-smoothness of the max function, the equivalence has little computational benefit. We smooth it by adding another term that penalizes deviations from a prior estimate of the  $\lambda$  parameters.

$$\min_x \max_{\lambda} f(x) + \lambda^t \mathbf{c}(x) + \frac{\|\lambda - \bar{\lambda}\|^2}{2\mu}$$


$$\lambda = \bar{\lambda} + \mu \mathbf{c}(x)$$

Maximization w.r.t.  $\lambda$   
is now easy

# ALM: Some intuition – inequality constraints

$$\begin{array}{ll} \min_x f(x) \\ \text{s.t. } \forall i \in I, c_i(x) \geq 0 \end{array} = \min_x \max_{\lambda \geq 0} f(x) - \lambda^t \mathbf{c}(x)$$

$\mathbf{c}(x) = (c_1(x), c_2(x), \dots, c_{|I|}(x))$

$$\min_x \max_{\lambda} f(x) + \lambda^t \mathbf{c}(x) + \frac{\|\lambda - \bar{\lambda}\|^2}{2\mu} \quad \longrightarrow \quad \lambda = \max(\bar{\lambda} + \mu \mathbf{c}(x), 0)$$

Maximization w.r.t.  $\lambda$   
is now easy



# Algorithm for Robust PCA

- In our case, we seek to optimize:

$$l(L, S, Y) = \|L\|_* + \lambda \|S\|_1 + \underbrace{\langle Y, M - L - S \rangle}_{\text{Lagrange matrix}} + \frac{\mu}{2} \|M - L - S\|_F^2.$$

- Basic algorithm:

$$(L_k, S_k) = \arg \min_{(L, S)} l(L, S, Y_k), Y_{k+1} = Y_k + \mu(M - L_k - S_k)$$

$$\arg \min_S l(L, S, Y) = \mathcal{S}_{\lambda\mu^{-1}}(M - L + \mu^{-1}Y).$$

Update of **S** using  
soft-thresholding

$$\mathcal{S}_\tau[x] = \text{sgn}(x) \max(|x| - \tau, 0)$$

$$\arg \min_L l(L, S, Y) = \mathcal{D}_{\mu^{-1}}(M - S + \mu^{-1}Y).$$

Update of **L** using  
singular-value  
soft-thresholding

$$\mathcal{D}_\tau(X) = U \mathcal{S}_\tau(\Sigma) V^* \quad X = U \Sigma V^*$$

## Alternating Minimization Algorithm for Robust PCA

- 1: **initialize:**  $S_0 = Y_0 = 0, \mu > 0$ .
  - 2: **while** not converged **do**
  - 3:     compute  $L_{k+1} = \mathcal{D}_{\mu^{-1}}(M - S_k + \mu^{-1}Y_k)$ ;
  - 4:     compute  $S_{k+1} = \mathcal{S}_{\lambda\mu^{-1}}(M - L_{k+1} + \mu^{-1}Y_k)$ ;
  - 5:     compute  $Y_{k+1} = Y_k + \mu(M - L_{k+1} - S_{k+1})$ ;
  - 6: **end while**
  - 7: **output:**  $L, S$ .
-

# (Compressive) Low Rank Matrix Recovery

# Low rank matrix **recovery**

- Note: the third set of experiments in the section on matrix completion are not really an example of matrix completion.
- They are an example of a related problem: low rank matrix recovery, defined as:

$$\mathbf{y} = \mathcal{A}(\mathbf{M}), \mathbf{M} \in R^{n_1 \times n_2}, \mathbf{y} \in R^m, m \leq n_1 n_2$$

$\mathcal{A}$  = linear operator acting/map on  $\mathbf{M}$

Retrieve  $\mathbf{M}$  given  $\mathcal{A}, \mathbf{y}$

# Low rank matrix **recovery**

$$\mathbf{y} = \mathcal{A}(\mathbf{M}), \mathbf{M} \in R^{n_1 \times n_2}, \mathbf{y} \in R^m, m \leq n_1 n_2$$

$\mathcal{A}$  = linear operator/map acting on  $\mathbf{M}$

Retrieve  $\mathbf{M}$  given  $\mathcal{A}, \mathbf{y}$

*NOTE:*

$$y_i = \text{vec}(\mathbf{A}_i) \bullet \text{vec}(\mathbf{M}) = \text{trace}(\mathbf{A}_i^T \mathbf{M}), \mathbf{A}_i \in R^{n_1 \times n_2}$$

$\{\mathbf{A}_i\}$  = set of matrices constituting the linear map  $\mathcal{A}$

EQUIVALENTLY :

$$\mathbf{y} = \mathbf{A} \text{vec}(\mathbf{M}), \mathbf{A} \in R^{m \times n_1 n_2}$$

# Low rank matrix recovery: restricted isometry property (RIP) for linear maps

- We consider  $\mathcal{A}$  to be a linear map from  $\mathbb{R}^{n_1 \times n_2}$  to  $\mathbb{R}^m$ .
- For integer  $r$  from 1 to  $\min(n_1, n_2)$ , the restricted isometry constant (RIC) of  $\mathcal{A}$  of order  $r$  is the smallest number  $\delta_r$  such that
$$(1 - \delta_r) \|\mathbf{M}\|_F \leq \|\mathcal{A}(\mathbf{M})\|_2 \leq (1 + \delta_r) \|\mathbf{M}\|_F$$
for all matrices  $\mathbf{M}$  of rank at the most  $r$ .

# Low rank matrix recovery: Matrix restricted isometry property (RIP)

- What linear maps obey RIP?
- The following matrices obey RIP:
  - ✓ Zero-mean iid Gaussian random matrices with variance  $1/m$
  - ✓  $(+1/-1)/\sqrt{m}$  Bernoulli random matrices

# Theorem 1 for low-rank matrix recovery

- If matrix  $\mathbf{A}$  is such that  $\delta_{2r}(\mathbf{A}) < 1$ , then the equation  $\mathbf{A} \text{vec}(\mathbf{M}) = \mathbf{y}$  has a unique solution where  $\mathbf{M}$  is a matrix of rank at the most  $r$ .
- Proof by contradiction:

$$\text{Avec}(\mathbf{M}) = \text{Avec}(\mathbf{M}_1) = \mathbf{y}$$

$$\therefore \mathbf{A}(\text{vec}(\mathbf{M}) - \text{vec}(\mathbf{M}_1)) = \mathbf{0}$$

But  $(\mathbf{M} - \mathbf{M}_1)$  has rank at most  $2r$

$$\therefore 0 = \|\mathbf{A}(\text{vec}(\mathbf{M}) - \text{vec}(\mathbf{M}_1))\|_2 \geq (1 - \delta_{2r}) \|\mathbf{M} - \mathbf{M}_1\|_F$$

$\therefore$  contradiction

Recht et al, "Guaranteed Minimum-Rank Solutions of Linear Matrix Equations via Nuclear Norm Minimization"



# Theorem 2 for low-rank matrix recovery

- If matrix  $\mathbf{A} \in \mathbb{R}^{m \times n_1 n_2}$  is such that  $\delta_{5r}(\mathbf{A}) < 0.1$  for  $r \geq 1$  and  $\mathbf{M} \in \mathbb{R}^{n_1 \times n_2}$  has rank at most  $r$ , then the solution to the following optimization problem has a unique solution:

$$(Q0) : \mathbf{M}^* = \min_{\mathbf{M}} \|\mathbf{M}\|_*$$

subject to

$$\mathbf{y} = \text{Avec}(\mathbf{M})$$

Recht et al, "Guaranteed Minimum-Rank Solutions of Linear Matrix Equations via Nuclear Norm Minimization"

# Theorem 2 for low-rank matrix recovery

- This theorem has extensions to the case where  $\mathbf{M}$  is approximately but not exactly of rank  $r$ .
- And also to the case where the measurements in vector  $\mathbf{y}$  are noisy.
- The proof of the result closely follows the proofs for compressive sensing recovery.

# Theorem 3 for low-rank matrix recovery

- Consider noisy measurements of the form:

$$\mathbf{y} = A \text{vec}(\mathbf{M}) + \boldsymbol{\eta}, \mathbf{A} \in R^{m \times n_1 n_2}, \mathbf{M} \in R^{n_1 \times n_2}, \mathbf{y} \in R^m, m \leq n_1 n_2, \|\boldsymbol{\eta}\|_2 \leq \varepsilon$$

- Let  $\mathbf{M}^*$  be the solution to the following problem:

$$\mathbf{M}^* = \arg \min \|\mathbf{M}\|_* \text{ such that } \|\mathbf{y} - A \text{vec}(\mathbf{M})\|_2 \leq \varepsilon$$

- Then if  $\delta_r(\mathbf{A}) < 1/3$ , we have:

$$\|\mathbf{M}^* - \mathbf{M}\|_F \leq C_1 \varepsilon + C_2 \|\mathbf{M} - \mathbf{M}_{(r)}\|_*$$

The best rank- $r$  approximation to  $\mathbf{M}$ .

$C_1$  and  $C_2$  are increasing functions of  $\delta_r(\mathbf{A})$  in the domain  $[0, 1/3]$ .

# Comments on Theorem 3

- It handles the case of noisy measurements as well as approximately (as opposed to exactly) low rank  $\mathbf{M}$ .
- There is demonstration of a case where a matrix with  $\delta_r(\mathbf{A}) = 1/3$  is unable to recover a set of low rank matrices  $\mathbf{M}$ .

# Compressive RPCA: Algorithm and an Application

Primarily based on the paper:

Waters et al, *“SpaRCS: Recovering Low-Rank  
and Sparse Matrices  
from Compressive Measurements”*, NIPS 2011

# Problem statement

- Let  $\mathbf{M}$  be a matrix which is the sum of low rank matrix  $\mathbf{L}$  and sparse matrix  $\mathbf{S}$ .
- We observed compressive measurements of  $\mathbf{M}$  in the following form:

$$\mathbf{y} = \mathcal{A}(\mathbf{L} + \mathbf{S}), \mathbf{L} \in R^{n_1 \times n_2}, \mathbf{S} \in R^{n_1 \times n_2}, \mathbf{y} \in R^m, m \leq n_1 n_2$$

$\mathcal{A}$  = linear operator acting/map on  $\mathbf{M}$

Retrieve  $\mathbf{L}, \mathbf{S}$  given  $\mathcal{A}, \mathbf{y}$

# Scenarios

- **M** could be a matrix representing a video – each column of **M** is a vectorized frame from the video.
- **M** could also be a matrix representing a hyperspectral image – each column is the vectorized form of a slice at a given wavelength.
- Robust Matrix completion – a special form of a compressive **L+S** recovery problem.

# Objective function: SpaRCS

$$(P1) \quad \min \quad \|\mathbf{y} - \mathcal{A}(\mathbf{L} + \mathbf{S})\|_2 \quad \text{subject to} \quad \text{rank}(\mathbf{L}) \leq r, \|\text{vec}(\mathbf{S})\|_0 \leq K.$$



Free parameters

SpaRCS = sparse and low rank decomposition via compressive sampling



# SparCS Algorithm

---

**Algorithm 1:**  $(\hat{\mathbf{L}}, \hat{\mathbf{S}}) = \text{SpaRCS}(\mathbf{y}, \mathcal{A}, \mathcal{A}^*, K, r, \epsilon)$

---

Initialization:  $k \leftarrow 1, \hat{\mathbf{L}}_0 \leftarrow \mathbf{0}, \hat{\mathbf{S}}_0 \leftarrow \mathbf{0}, \Psi_{\mathbf{L}} \leftarrow \emptyset, \Psi_{\mathbf{S}} \leftarrow \emptyset, \mathbf{w}_0 \leftarrow \mathbf{y}$

**while**  $\|\mathbf{w}_{k-1}\|_2 \geq \epsilon$  **do**

    Compute signal proxy:

$$\mathbf{P} \leftarrow \mathcal{A}^*(\mathbf{w}_{k-1})$$

    Support identification:

$$\hat{\Psi}_{\mathbf{L}} \leftarrow \text{svd}(\mathbf{P}; 2r); \hat{\Psi}_{\mathbf{S}} \leftarrow \text{supp}(\mathbf{P}; 2K)$$

    Support merger:

$$\tilde{\Psi}_{\mathbf{L}} \leftarrow \hat{\Psi}_{\mathbf{L}} \cup \Psi_{\mathbf{L}}; \tilde{\Psi}_{\mathbf{S}} \leftarrow \hat{\Psi}_{\mathbf{S}} \cup \Psi_{\mathbf{S}}$$

    Least squares estimation:

$$\mathbf{B}^{\mathbf{L}} \leftarrow \tilde{\Psi}_{\mathbf{L}}^\dagger(\mathbf{y} - \mathcal{A}(\hat{\mathbf{S}}_{k-1})); \mathbf{B}^{\mathbf{S}} \leftarrow \tilde{\Psi}_{\mathbf{S}}^\dagger(\mathbf{y} - \mathcal{A}(\hat{\mathbf{L}}_{k-1}))$$

    Support pruning:

$$(\hat{\mathbf{L}}_k, \Psi_{\mathbf{L}}) \leftarrow \text{svd}(\mathbf{B}^{\mathbf{L}}; r); (\hat{\mathbf{S}}_k, \Psi_{\mathbf{S}}) \leftarrow \text{supp}(\mathbf{B}^{\mathbf{S}}; K)$$

    Update residue:

$$\mathbf{w}_k \leftarrow \mathbf{y} - \mathcal{A}(\hat{\mathbf{L}}_k + \hat{\mathbf{S}}_k)$$

$$k \leftarrow k + 1$$

**end**

$$\hat{\mathbf{L}} = \hat{\mathbf{L}}_{k-1}; \hat{\mathbf{S}} = \hat{\mathbf{S}}_{k-1}$$

---

<https://papers.nips.cc/paper/4438-sparcs-recovering-low-rank-and-sparse-matrices-from-compressive-measurements.pdf>

Very simple to implement; but requires tuning of  $K, r$  parameters; convergence guarantees not established.

# Results: Phase transition

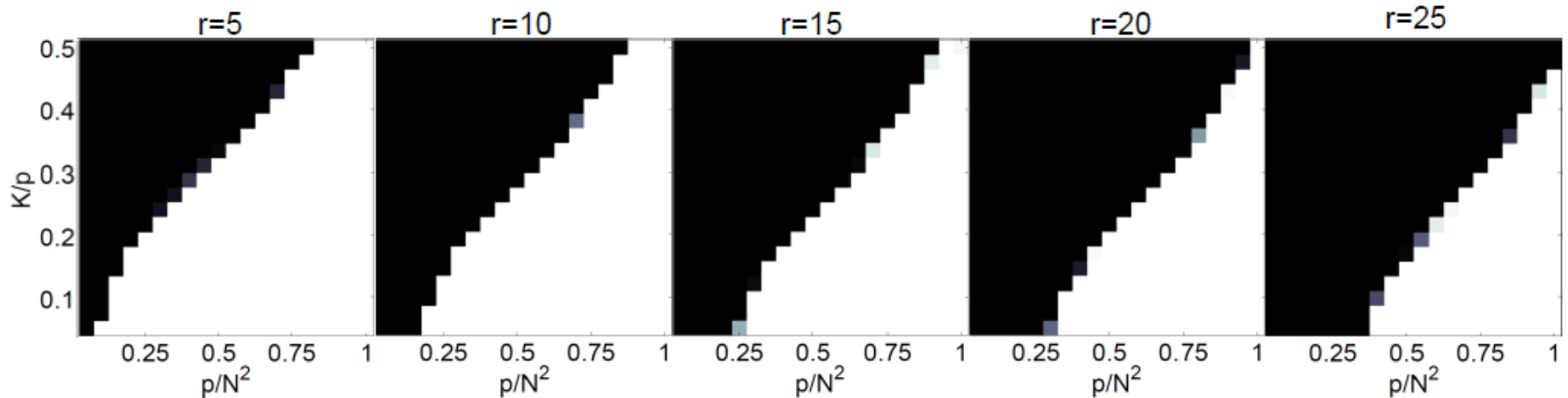


Figure 1: Phase transitions for a recovery problem of size  $N_1 = N_2 = N = 512$ . Shown are aggregate results over 20 Monte-Carlo runs at each specification of  $r$ ,  $K$ , and  $p$ . Black indicates recovery failure, while white indicates recovery success.

<https://papers.nips.cc/paper/4438-sparcs-recovering-low-rank-and-sparse-matrices-from-compressive-measurements.pdf>

# Results: Video CS

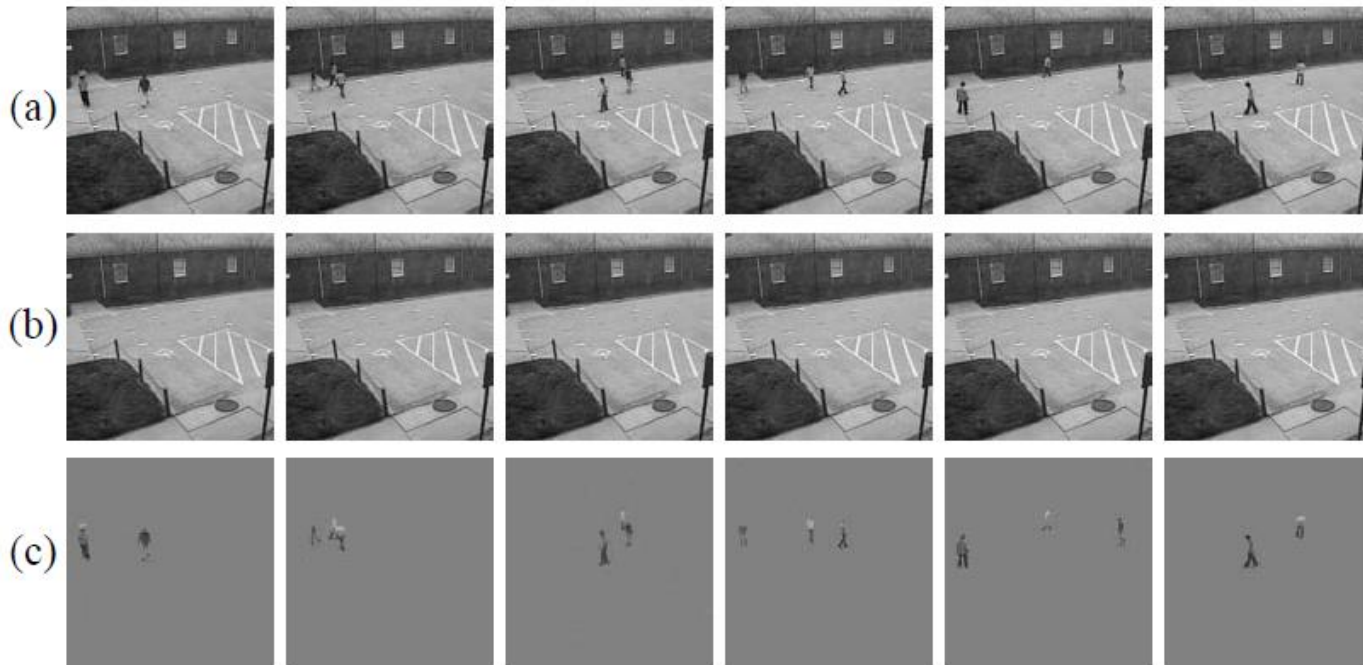


Figure 3: SpaRCS recovery results on a  $128 \times 128 \times 201$  video sequence. The video sequence is reshaped into an  $N_1 \times N_2$  matrix with  $N_1 = 128^2$  and  $N_2 = 201$ . (a) Ground truth for several frames. (b) Estimated low-rank component  $\mathbf{L}$ . (c) Estimated sparse component  $\mathbf{S}$ . The recovery SNR is 31.2 dB at the measurement ratio  $p/(N_1 N_2) = 0.15$ . The recovery is accurate in spite of the measurement operator  $\mathcal{A}$  working independently on each frame.

Follows Rice SPC model, independent compressive measurements on each frame of the matrix  $\mathbf{M}$  representing the video.

<https://papers.nips.cc/paper/4438-sparcs-recovering-low-rank-and-sparse-matrices-from-compressive-measurements.pdf>

# Results: Video CS



Figure 4: SpaRCS recovery results on a  $64 \times 64 \times 234$  video sequence. The video sequence is reshaped into an  $N_1 \times N_2$  matrix with  $N_1 = 64^2$  and  $N_2 = 234$ . (a) Ground truth for several frames. (b) Recovered frames. The recovery SNR is 23.9 dB at the measurement ratio of  $p/(N_1 N_2) = 0.33$ . The recovery is accurate in spite of the changing illumination conditions.

Follows Rice SPC model, independent compressive measurements on each frame of the matrix  $\mathbf{M}$  representing the video.

<https://papers.nips.cc/paper/4438-sparcs-recovering-low-rank-and-sparse-matrices-from-compressive-measurements.pdf>



# Results: Hyperspectral CS

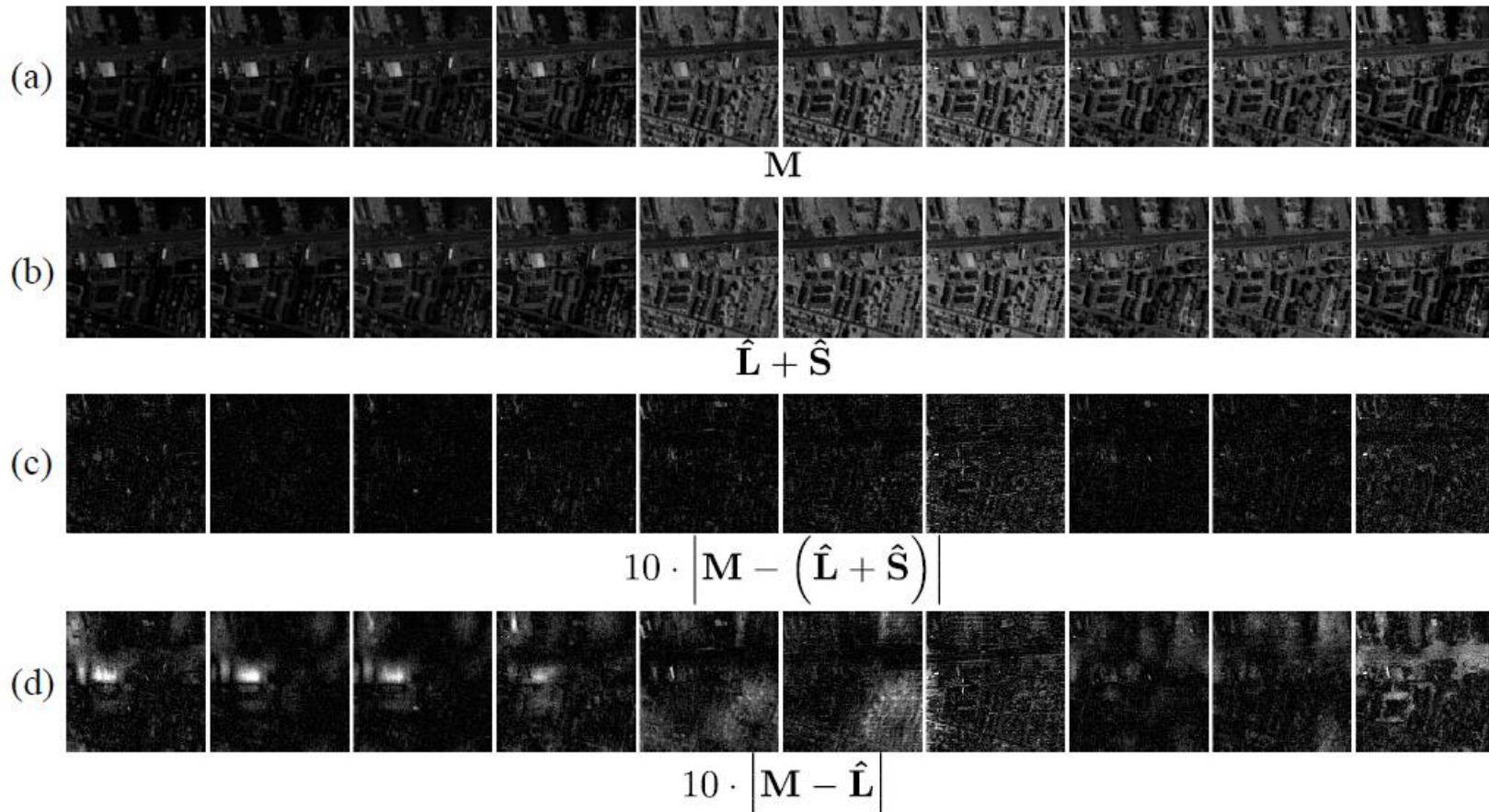


Figure 5: SpaRCS recovery results on a  $128 \times 128 \times 128$  hyperspectral data cube. The hyperspectral data is reshaped into an  $N_1 \times N_2$  matrix with  $N_1 = 128^2$  and  $N_2 = 128$ . Each image pane corresponds to a different spectral band. (a) Ground truth. (b) Recovered images. (c) Residual error using both the low-rank and sparse component. (d) Residual error using only the low-rank component. The measurement ratio is  $p/(N_1 N_2) = 0.15$ .

# Results: Robust matrix completion

<https://papers.nips.cc/paper/4438-sparcs-recovering-low-rank-and-sparse-matrices-from-compressive-measurements.pdf>

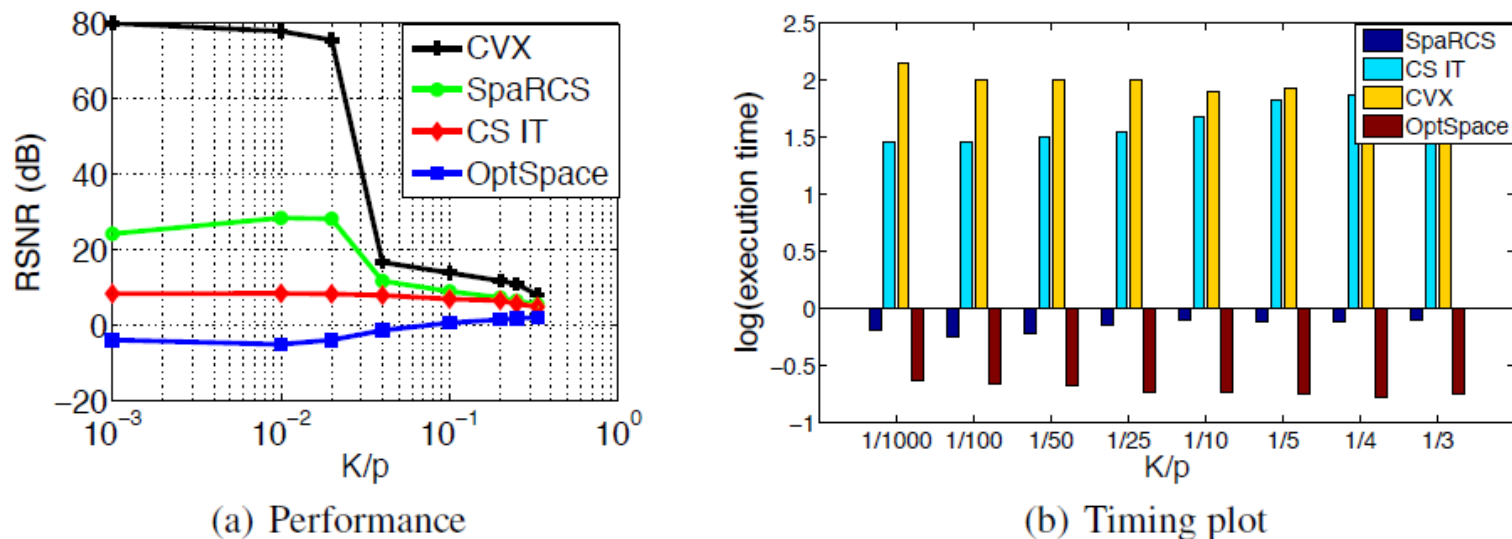


Figure 7: Comparison of several algorithms for the robust matrix completion problem. (a) RSNR averaged over 10 Monte-Carlo runs for an  $N \times N$  matrix completion problem with  $N = 128$ ,  $r = 1$ , and  $p/N^2 = 0.2$ . Non-robust formulations, such as OptSpace, fail. SpaRCS achieves performance close to that of the convex solver (CVX). (b) Comparison of convergence times for the various algorithms. SpaRCS converges in only a fraction of the time required by the other algorithms.

$$\min \|\mathbf{L}\|_* + \lambda \|\mathbf{s}\|_1 \quad \text{subject to } \mathbf{L}_\Omega + \mathbf{s} = \mathbf{y}$$

# Theorem for Compressive PCP

**Theorem 2.1 (Compressive PCP Recovery).** Let  $\mathbf{L}_0, \mathbf{S}_0 \in \mathbb{R}^{m \times n}$ , with  $m \geq n$ , and suppose that  $\mathbf{L}_0 \neq \mathbf{0}$  is a rank- $r$ ,  $\mu$ -incoherent matrix with

$$r \leq \frac{c_r n}{\mu \log^2 m}, \quad (2.4)$$

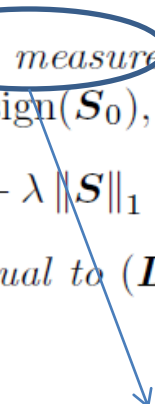
and  $\text{sign}(\mathbf{S}_0)$  is iid Bernoulli-Rademacher with nonzero probability  $\rho < c_\rho$ . Let  $Q \subset \mathbb{R}^{m \times n}$  be a random subspace of dimension

$$\dim(Q) \geq C_Q \cdot (\rho mn + mr) \cdot \log^2 m \quad (2.5)$$

distributed according to the Haar measure, probabilistically independent of  $\text{sign}(\mathbf{S}_0)$ . Then with probability at least  $1 - Cm^{-9}$  in  $(\text{sign}(\mathbf{S}_0), Q)$ , the solution to

$$\text{minimize} \quad \|\mathbf{L}\|_* + \lambda \|\mathbf{S}\|_1 \quad \text{subject to} \quad \mathcal{P}_Q[\mathbf{L} + \mathbf{S}] = \mathcal{P}_Q[\mathbf{L}_0 + \mathbf{S}_0] \quad (2.6)$$

with  $\lambda = 1/\sqrt{m}$  is unique, and equal to  $(\mathbf{L}_0, \mathbf{S}_0)$ . Above,  $c_r, c_\rho, C_Q, C$  are positive numerical constants.



Q is obtained from the linear span  
of different independent  $N(0,1)$   
matrices with iid entries

Wright et al, "Compressive Principal Component Pursuit"

<http://yima.csl.illinois.edu/psfile/CPCP.pdf>

# Theorem for Compressive PCP

Notion of  $\mu$ -incoherent L0:

$$L_0 = U \Sigma V^*$$

$$\forall i \quad \|U^* e_i\|_2^2 \leq \frac{\mu r}{m}, \quad \forall j \quad \|V^* e_j\|_2^2 \leq \frac{\mu r}{n}, \quad \text{and} \quad \|UV^*\|_\infty \leq \sqrt{\frac{\mu r}{mn}}.$$

Definition of  $Q$ :

$$Q : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^q$$

$$Q[M] = (\langle Q_1, M \rangle, \dots, \langle Q_q, M \rangle)^* \in \mathbb{R}^q.$$

Wright et al, "Compressive Principal Component Pursuit"

<http://yima.csl.illinois.edu/psfile/CPCP.pdf>



# Summary

- Low rank matrix completion: motivation, key theorems, numerical results
- Algorithm for low rank matrix completion
- Robust PCA
- (Compressive) low rank matrix recovery
- Compressive RPCA
- Several papers linked on moodle