Efficient Computation of Robust Low-Rank Matrix Approximations in the Presence of Missing Data using the $L_1$ Norm

Anders Eriksson, Anton van den Hengel

School of Computer Science
University of Adelaide, Australia

Presented by

Wang Yuxiang

Department of Electrical and Computer Engineering, National University of Singapore
Outline

• Introduction

• Previous work
  – ALP & AQP
  – Alternating Least Square
  – Gauss Newton/Levenberg Marquadt
  – Wiberg L2 Algorithm

• Proposed algorithm
  – A detour to LP
  – Wiberg L1 Algorithm

• Experiments
  – Synthesized data
  – The Dinosaur sequence
  – Comparison with Rosper

• Conclusion
Introduction: Problem formulation

- Low-rank matrix approximation in the presence of missing data.

\[
\min_{U,V} \| \hat{W} \otimes (Y - UV) \|, \\
Y \in \mathbb{R}^{m \times n} \quad U \in \mathbb{R}^{m \times r} \quad V \in \mathbb{R}^{r \times n} \quad \hat{W} \in \mathbb{R}^{m \times n}
\]
Introduction: Background

- Many applications such as SfM, shape from varying illumination (photometric stereo).
- Simple SVD (ML minimization of L2 Norm) is not applicable due to missing data.
- L2 norm is sensitive to outliers.
- L1 norm is used to reduce sensitivity to outliers.
Introduction: Advantage of L1 Nnorm

Figure 1. *Fit a line to 10 given data points. The two data points on upper-left are outliers.*

Extracted from [13]
Introduction: Challenges of L1

- Non-smooth (hence non-differentiable)
- Non-convex due to rank constraint and missing data. (Is this related to adoption of L1 norm?)
- More computational demanding than L2 (we’ll see)
Key contribution of this paper

- Extends Wiberg algorithm to L1 norm
- Proposed algorithm is more efficient than other algorithms that deal with same problem.
- Address non-smoothness and computational requirement using LP.
Previous Works

[5] Buchanan, a quantitative survey

[16] Okatani 岡谷, Reintroduce Wiberg to Computer Vision (more to come)

[13] Ke&Kanade, ALP and AQP (state-of-art as claimed, explained in next slide)

[9] De la Torre&Black a robust version of PCA based on Huber distance.

Previous Works: Ke & Kanade

• Ke&Kanade, Robust L1 norm factorization in the presence of outliers and missing data by alternative convex programming, CVPR2005

• Alternative Convex Programming

\[
V^{(t)} = \arg \min_{V} \| M - U^{(t-1)}V^\top \|_{L_1} \tag{8a}
\]

\[
U^{(t)} = \arg \min_{U} \| M - UV^{(t)}^\top \|_{L_1} \tag{8b}
\]
Previous Works: Ke & Kanade

- **Alternated Linear Programming:**

\[
E(V) = \| M - U^{(t-1)}V^T \|_{L_1} = \sum_{j=1}^{n} \| m_j - U^{(t-1)}v_j \|_1 \tag{9}
\]

where \( m_j \) is the \( j \)-th column of \( M \), \( v_j \) is the \( j \)-th column of \( V^T \). The problem of Eq. (8a) is therefore decomposed into \( n \) independent small sub-problems, each one optimizing \( v_j \):

\[
v_j = \arg \min_x \| U^{(t-1)}x - m_j \|_1 \tag{10}
\]

The *global* optimal solution of Eq. (10) can be found by the following linear program (LP):

\[
\min_{x,t} 1^T t
\]

\[
s.t. \quad -t \leq U^{(t-1)}x - m_j \leq t \tag{11}
\]
Previous Works: Ke & Kanade

- Alternated Quadratic Programming:

  Huber Norm:

  \[ \rho(e) = \begin{cases} 
  \frac{1}{2}e^2, & \text{if } |e| \leq \gamma \\
  \gamma|e| - \frac{1}{2}\gamma^2, & \text{if } |e| > \gamma 
  \end{cases} \]

  \[ v_j = \arg \min_x \rho(U^{(t-1)}x - m_j) \]  \hspace{1cm} (12)

Since Huber M-estimator is a differentiable convex function, Eq. (12) can be converted to a convex quadratic programming (QP) problem whose *global* minimum can be computed efficiently [17]:

\[ \min_{x,z,t} \frac{1}{2}\|z\|_2^2 + \gamma 1^T t \]

\[ \text{s.t. } -t \leq U^{(t-1)}x - m_j - z \leq t \]  \hspace{1cm} (13)
Previous Works: Ke & Kanade

Remarks:
1. Handle missing data by dropping constraints in LP and QP formulation.

2. Result in convex sub problem, but solution might not be good for the original problem.
Notations:

• Small case means “vec” operator.
• $W = \text{diag}(\hat{W})$, so $Wy$ means the masked $\text{vec}(Y)$.
• $\otimes$: Kronecker Product

\[
A \otimes B = \begin{bmatrix}
a_{11}B & \cdots & a_{1n}B \\
\vdots & \ddots & \vdots \\
a_{m1}B & \cdots & a_{mn}B
\end{bmatrix}.
\]
Property of Kronecker Product

\[(B^\top \otimes A) \text{vec}(X) = \text{vec}(AXB) = \text{vec}(C).\]

- Assume V is fixed, U is unknown,
  \[\text{vec}(UV) = \text{vec}(IUV) = (V^\top \otimes I)\text{vec}(U);\]
- Assume U is fixed and V is unknown,
  \[\text{vec}(UV) = \text{vec}(UVI) = (I \otimes U)\text{vec}(V);\]
Alternated least square (ALS)

- So \( \min_{U,V} \| \hat{W} \odot (Y - UV) \| \), is equivalent to:

\[
\min_v \| Wy - W (I_n \otimes U) v \|_2^2, = \| Wy - G(U) v \|_2^2
\]

\[
\min_u \| Wy - W (V^T \otimes I_m) u \|_2^2, = \| Wy - F(V) u \|_2^2
\]

\( f = Fu - Wy = Gv - Wy; \)

\( 1/2f^Tf = \Phi(U,V) \)

Take partial derivative of \( \Phi \) and assign to 0, we get:

\( u = (F^TF)^{-1}F^T (Wy); \ v = (G^TG)^{-1}G^T (Wy); \)
Gauss-Newton Method

• Let $x=[u^T \ v^T]$, $\Phi(U,V) = \frac{1}{2}f^Tf$

• Newton’s method: To attain $\partial \Phi/\partial x=0$ using the step $\delta$, satisfying:

$$\frac{\partial^2 \Phi}{\partial x^2} \Delta x + \frac{\partial \Phi}{\partial x} = 0$$

(*)

• Calculate the first and second derivative:

$$\frac{\partial \Phi}{\partial x} = f^T \frac{\partial f}{\partial x}$$

$$\frac{\partial^2 \Phi}{\partial x^2} = (\frac{\partial f}{\partial x})^T \frac{\partial f}{\partial x} + f^T \frac{\partial^2 f}{\partial x^2} \approx (\frac{\partial f}{\partial x})^T \frac{\partial f}{\partial x}$$

• Note that $\frac{\partial f}{\partial x} = [\frac{\partial f}{\partial u} \ \frac{\partial f}{\partial v}] = [F \ G]$

Equation (*) now becomes a $Ax=b$ problem, which can be solved by least square.
Wiberg Algorithm

• Fixing part of the parameters AND apply the idea of Newton’s method with Gauss-Newton Assumption.
• By doing so (fixing V for example), the algorithm becomes more efficient.
• Let $\Phi(U,V) = \psi(U,V(U)) = 1/2g^Tg$
  where $g(u)=f(u,v(u))$ is a function of $u$ only.
• Now try to use Newton’s method:
  \[ \frac{\partial^2 \psi}{\partial u^2} \Delta u + \frac{\partial \psi}{\partial u} = 0 \]
  We need to know the first and second derivative of $\psi$ w.r.t. $u$. 

Wiberg Algorithm

- Again, using Gauss-Newton Approx, we have:
  \[
  \frac{\partial \psi}{\partial u} = g^T \frac{\partial g}{\partial u}
  \]
  \[
  \frac{\partial^2 \psi}{\partial u^2} = (\frac{\partial g}{\partial u})^T \frac{\partial g}{\partial u} + g^T \frac{\partial^2 g}{\partial u^2} \approx (\frac{\partial g}{\partial u})^T \frac{\partial g}{\partial u}
  \]
- Using chain rule:
  \[
  \frac{\partial g}{\partial u} = \frac{\partial f}{\partial u} + \frac{\partial f}{\partial v} \frac{\partial v}{\partial u} = F + G(\frac{\partial v}{\partial u})
  \]
- At optimal \((u,v)\): \(\frac{\partial \Phi}{\partial u} = 0\) AND \(\frac{\partial \Phi}{\partial v} = 0\) (KKT)
  \(\frac{\partial \Phi}{\partial u} = \frac{\partial \psi}{\partial u}\) will be enforce to 0 by Gauss-Newton.
  \(\frac{\partial \Phi}{\partial v} = 0\) is irrelevant to \(u\), so:
  \[
  \frac{\partial^2 \Phi}{\partial v \partial u} = \frac{\partial}{\partial u} \left( \frac{f^T \partial f}{\partial v} \right) = \frac{\partial}{\partial u} \left( f^T G \right) = (\frac{\partial f}{\partial v} * \frac{\partial v}{\partial u} + \frac{\partial f}{\partial u})^T G + f^T(\frac{\partial G}{\partial u}) \approx 0
  \]
Wiberg Algorithm

- \( (\partial f/\partial v \times \partial v/\partial u + \partial f/\partial u)^T G = 0 \)
  Take Transpose both side:
  \( G^T (G \partial v/\partial u + F) = 0 \)
- \( \partial v/\partial u = -(G^T G)^{-1} G^T F \)
- Substitute into \( \partial g/\partial u = F + G(\partial v/\partial u) = (I - G(G^T G)^{-1} G^T) F \)
  Let \( Q_G = I - G(G^T G)^{-1} G^T \)
- We obtained the step \( \Delta u \) for Gauss-Newton update in:
  Minimize \( || - Q_G W y + Q_G F \Delta u || \)
  This is a least square problem. As derived in Okatani Paper [16].
Insufficient Rank of $Q_{GF}$

• Okatani proved that this $Q_{GF}$ have dimension $(m-r)r$ instead of full rank $mr$.

• So further constraints that $|| \Delta u ||$ is minimized should be used to uniquely determine a $\Delta u$.

• Yet, in this Eriksson paper, it is not mentioned. (corresponds to the Insufficient Rank of Jacobian)
Connection to Linearized Model

Consider

$$\text{Minimize } || - Q_G Wy + Q_G F \Delta u ||$$

Substitute in the value of $Q_G = I - G(G^T G)^{-1} G^T$

We have

$$\text{Minimize} || - Wy + G(G^T G)^{-1} G^T Wy + Q_G F \Delta u ||$$

Recall that $G(G^T G)^{-1} G^T Wy = Gv^*(u)$ and $\frac{\partial g}{\partial u} = Q_G F$

We may define function $g' = g + Wy = Gv^*(u)$,

Since $Wy$ is const. so $\frac{\partial g'}{\partial u} = \frac{\partial g}{\partial u}$
Connection to Linearized Model

• Now it becomes
  \[
  \text{Minimize } \| -W y + g'(u_k) + \frac{\partial g'}{\partial u} \Delta u \| 
  \]

• This corresponds to Equation (8) in the paper and \( J_k = \frac{\partial g'}{\partial u} \big|_{u=u_k} = Q_G F \).
  (Btw there’s a missing term in Equation (8))

• We have showed that Gauss-Newton update is equivalent to such linearized model.
Proposed algorithm: Wiberg L1

• So using the same “Linearized Model” argument, the L2 Wiberg model can be extended to L1.

\[ \min_{U,V} \| \hat{W} \odot (Y - UV) \|_1. \]
Detour: Linear Programming

- Canonical form of an LP

\[
\min_{x \in \mathbb{R}^n} \quad c^T x \\
\text{s.t.} \quad Ax = b \\
\quad x \geq 0
\]

- w.o.l.g, \( A \) (m*n) is of full rank m.
- No inequality constraints because we can add slack variables.
Basic Solution and Non-basic Solution

• Reorder $A$ to get $A=[B \ N]$ where $B$ is $m \times m$ non-singular.

• $x=[x_B \ x_N]$ and $x_N=0$, then $x$ is b.s.

• If $x_B \geq 0$, $x$ is called basic feasible solution (b.f.s.), if $c^T x$ is optimal, then optimal basic solution.

• **Fundamental Thm of LP**: If an LP is feasible, then exists a optimal b.f.s.
Simplex Method
Simplex Method

- Pivot operation to go from one vertex to another until optimal (in finite steps).
- Each vertex represents a set of b.f.s., pivot changes columns between B and N and find a new b.f.s.
- Can be done efficiently using any commercial solver.
Sensitivity Analysis of LP

• How does optimal solution change if we change constraints ($A$ and $b$) or objective function ($c$) locally?

• Thm 3.2:

\[
\frac{\partial x_B^*}{\partial B} = -(x_B^*)^T \otimes B^{-1} \tag{13}
\]

\[
\frac{\partial x_B^*}{\partial N} = 0 \tag{14}
\]

\[
\frac{\partial x_B^*}{\partial b} = B^{-1} \tag{15}
\]

\[
\frac{\partial x_N^*}{\partial A} = \frac{\partial x_N^*}{\partial b} = 0. \tag{16}
\]
The use of sensitivity analysis in Wiberg L1

1. Fix \( U = U_k \), calculate the optimal L1 Norm w.r.t. \( V \) using LP, and get \( V_k = V^* \).

2. Based on the sensitivity analysis, calculate the gradient of \( V^* \) w.r.t. \( U \) at \( U_k \), hence linearize \( \Phi(U) \) at \( U_k \) and calculate the optimal L1 Norm w.r.t. a change step \( \delta \) of \( U \) using LP again.

3. Then update \( U \) and proceed in a gradient descent manner.
Proposed Algorithm: Wiberg L1

\[ v^*(U) = \arg \min_v \|W y - W (I_n \otimes U) v\|_1, \]

\[ u^*(V) = \arg \min_u \|W y - W (V^T \otimes I_m) u\|_1, \]

\[ f = F(V)u - Wy = G(U)v - Wy \]

\[ \min_U f(U) = \|W y - W U V^*(U)\|_1 = \]

\[ = \|W y - \phi_1(U)\|_1. \]

\[ \Phi_1(u) = \text{vec}(W U V^*(U)) \]

\[ = F(V)u = G(U)v = G(u)v^*(u) = F(v^*(u))u \]
LP to find \( v^* \) given any \( U \)

- Given a \( U_k \), we can solve for \( v^* \) that minimize the L1 norm using the following LP. (Simplex will do.)

\[
\begin{align*}
\min_{v^+, v^-, t, s} & \quad 0 \quad 0 \quad 1^T \quad 0 \\
\text{s.t.} & \quad \begin{bmatrix} -G(U) & G(U) & -I & I \end{bmatrix} \begin{bmatrix} v^+ \\ v^- \\ t \\ s \end{bmatrix} = \begin{bmatrix} -Wy \\ Wy \\ b \end{bmatrix} \\
& \quad v^+, v^-, t, s \geq 0 \\
& \quad v^+, v^- \in \mathbb{R}^{rn}, \ t \in \mathbb{R}^{mn}, \ s \in \mathbb{R}^{2mn}.
\end{align*}
\]
Given $v^*$, linearize the model and find the optimal $\Delta u$

- In order to do that, we must calculate $J(U)$
  \[
  \frac{\partial \Phi_1}{\partial u} = \frac{\partial \Phi_1}{\partial u} + \frac{\partial \Phi_1}{\partial v^*} \frac{\partial v}{\partial u}
  \]

  \[
  = J(U) = F(V) + G(U) \frac{\partial v^*}{\partial U}.
  \]

- $F$ and $G$ is constant or known,
  \[
  \frac{\partial v^*}{\partial U} = \frac{\partial v^*+}{\partial U} - \frac{\partial v^*^-}{\partial U}
  \]
Given $v^*$, linearize the model and find the optimal $\Delta u$

$$\frac{\partial z^*}{\partial U} = \left[ \begin{array} {c} \frac{\partial v^*+}{\partial U} \\ \frac{\partial v^*-}{\partial U} \\ \frac{\partial t^*}{\partial U} \\ \frac{\partial s^*}{\partial U} \end{array} \right] = \frac{\partial z^*}{\partial B} \frac{\partial B}{\partial G} \frac{\partial G}{\partial U}$$

$$\frac{\partial G}{\partial U} = (I_{nr} \otimes W) (I_n \otimes T_{r,n} \otimes I_m) \left( \text{vec}(I_n) \otimes I_{mr} \right)$$

$$\frac{\partial B}{\partial G} = \frac{\partial (AQ)}{\partial G} = (Q^T \otimes I_{2mn}) \frac{\partial A}{\partial G} =$$

$$= (Q^T \otimes I_{2mn}) \left[ \frac{\partial}{\partial G} \begin{bmatrix} -G & G \\ G & -G \end{bmatrix} \right] =$$

$$= (Q^T \otimes I_{2mn}) \left[ \frac{\partial}{\partial G} \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix} \otimes G \right]$$

$$\frac{\partial z^*}{\partial B} = Q \frac{\partial z_B^*}{\partial B} = -Q \left( (z_B^*)^T \otimes B^{-1} \right)$$
Minimizing L1 norm over $\Delta u$

- To find the optimal $\Delta u$, we again want to minimize the L1 norm.

\[
\min_{\delta} \|Wy - J(U)(\delta - u)\|_1
\]

- Again, there’s a typo, a term is missing here at (41)(42) in the paper. Also, the $J(u_k)u$ term makes no sense at all. Correct formulation is:

\[
\min. \|Wy - \Phi_1(u_k) - J(u_k)(\Delta u)\|
\]
LP to find $\Delta u$ and minimize L1 norm

- Note the trust region defined here.
- Only within a small step, the linearity assumption is true.

\[
\begin{align*}
\min_{\delta, t} & \quad \begin{bmatrix} 0 & 1^T \end{bmatrix} \begin{bmatrix} \delta \\ t \end{bmatrix} \\
\text{s.t.} & \quad \begin{bmatrix} -J(U) - I & J(U) - I \end{bmatrix} \begin{bmatrix} \delta \\ t \end{bmatrix} = \begin{bmatrix} -(W_y - W \text{vec}(UV^*)) \\ W_y - W \text{vec}(UV^*) \end{bmatrix} \\
& \quad ||\delta||_1 \leq \mu \\
& \quad \delta \in \mathbb{R}^{mr}, \ t \in \mathbb{R}^{mn}.
\end{align*}
\]

- Again, this is solved by Simplex method.
Algorithm 1: $L_1$-Wiberg Algorithm

```
input : $U_0$, $1 > \eta_2 > \eta_1 > 0$ and $c > 1$

$k = 0$

repeat

3. Compute the Jacobian of $\phi_1 = J(U_k)$ using (37)-(40);

4. Solve the subproblem (43)-(46) to obtain $\delta_k^*$;

5. Let $gain = \frac{f(U_k) - f(U_k + \delta^*)}{\tilde{f}(U_k) - \tilde{f}(U_k + \delta^*)}$;

6. if $gain \geq \epsilon$ then

7. $U_{k+1} = U_k + \delta^*$;

8. end

9. if $gain < \eta_1$ then

10. $\mu = \eta_1 \|\delta^*\|_1$

11. end

12. if $gain > \eta_2$ then

13. $\mu = c\mu$

14. end

15. $k = k + 1$;

16. until convergence;
```
Experiments

• The paper presents comparison between proposed algorithm and ALP & AQP presented in [13] over synthetic data.

• There is also an application in SfM using the Dinosaur Sequence and compare to Wiberg L2.

• Results are pretty presentable in terms accuracy of recovery and speed.
Synthetic Data Matrix

- $m = 7; n = 12; r = 3$; *(very small!)*
- 20% Random Distributed Missing Entries
- 10% Random Distributed Sparse outlier noise $\text{Uni}(-5,5)$
- Tested over 100 data matrices to obtain a statistic.
Figure 1. A histogram representing the frequency of different magnitudes of error in the estimate generated by each of the methods.

[Histogram: Frequency vs. Error]
Speed of convergence: Residual vs. Iteration

Figure 2. Plots showing the norm of the residual at each iteration of two randomly generated tests for both the $L_1$ Wiberg and alternated quadratic programming algorithms. [Residual norm vs. Iteration]
Speed of convergence: log error vs. Iteration

Figure 3. A plot showing the convergence rate of the alternated quadratic programming and $L_1$-Wiberg algorithms over 100 trials. The error are presented on a logarithmic scale. [Log Error vs. Iteration]
Table of Results

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Error ($L_1$)</td>
<td>4.60</td>
<td>2.29</td>
<td>1.01</td>
</tr>
<tr>
<td>Execution Time (sec)</td>
<td>0.16</td>
<td>93.57</td>
<td>1.51</td>
</tr>
<tr>
<td># Iterations</td>
<td>4.72</td>
<td>177.64*</td>
<td>21.77</td>
</tr>
<tr>
<td># LP/QP solved</td>
<td>9.44</td>
<td>355.28*</td>
<td>24.13</td>
</tr>
<tr>
<td>Time per LP/QP</td>
<td>0.016</td>
<td>0.264*</td>
<td>0.061</td>
</tr>
</tbody>
</table>

* The alternated QP algorithm was terminated after 200 iterations and 400 solved quadratic programs.

Table 1. *The averaged results from running 100 synthetic experiments.*

- ALP has poor results in terms of error, though speed is almost real time.
- AQP has long running time and moderate performance.
- Wiberg outperforms others in average error, and is considerably fast.
Structure from Motion

• 319 features are tracked over 36 views
• Uni[-50,50] outliers are artificially added into the 10% of tracked points.
• Unknown sample rate
Data matrix of Dinosour Sequence

Generated using the data provided by Oxford Visual Geometry Group: There are 4983 feature tracks instead of 319.
http://www.robots.ox.ac.uk/~vgg/data1.html
Comparison of L1 and L2 Wiberg completion results

Figure 4. Resulting residuals using the standard Wiberg algorithm (top), and our proposed $L_1$-Wiberg algorithm (bottom).
Result table and Reconstructed point cloud

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>RMS Error of Inliers</td>
<td>-</td>
<td>2.029</td>
<td>0.862</td>
</tr>
<tr>
<td>Execution Time</td>
<td>&gt;4 hrs</td>
<td>3 min 2 sec</td>
<td>17 min 44 sec</td>
</tr>
</tbody>
</table>

Table 2. Results from the dinosaur sequence with 10% outliers.

Figure 5. Images from the dinosaur sequence, and the resulting reconstruction using our proposed method.
Critique on this SfM attempt

• Too little information is provided how they did this SfM. What’s the sample rate? The factorization to U and V is not unique as mentioned earlier.

• The code provided doesn’t work for 72*319 at all. It breaks down at 20*70…

• What residual is used in Figure 4 is not clear L1 or L2.
Comparison with Rosper

• Rosper: Lee Choon Meng’s explicit rank constraint low-rank optimizer

\[
\min \frac{1}{2} \| \mathcal{A}(W) + \mathcal{A}(E) - \mathcal{A}(\hat{W}) \|_F^2 + \mu \| E \|_{2,1} \\
\text{s.t. } \text{rank}(W) \leq r
\]

• Solving by Proximal Gradient (no A due to non-convexity)

• Optimize W and E concurrently
Compared algorithms

• **Wiberg L1**: unoptimized version of code provided by [Anders Eriksson](https://www.anders-eriksson.com). Slow and not scalable. (Not working for matrices as small as 20*80)

• **Original Wiberg (L2)**: provided by [Okatani](https://www.okatani.com), citation [16]. Features checking if a data matrix is well-posed for the algorithm. (If $Q_FG$ is of its maximum rank $(n-r)r$)

• These two algorithms both result in non-unique $U$ and $V$. In order to compare, we only compare $W=U*V$. 
Difference in Error Model

- Wiberg L1 minimizes L1 Norm, which handles sparse outliers in data matrix.
- Wiberg L2 minimizes L2 Norm, which in theory is optimal (MLE) if the noise is Gaussian distributed.
- Rosper models noise more explicitly. The first term handles dense Gaussian noise while the second term handles sparse outliers.
Experiment design

- Data matrix dimension: 20*30
- Each element of U and V fall within [0,1]
- Sample rate: ~40%
- Uniformly sampled
- Noise of various scale:
  - Uniform noise to represent sparse outlier
  - Gaussian noise to represent dense noise
An illustration of the sampled data matrix
Experiment 3: 0.3 Error
Experiment 5: 0.5 Error
Experiment 6: 0.7 Error
Experiment 4: 0 Error
Different error rate

L1 Error Norm over different error rate

- Wiberg L1
- Wiberg L2
- Rosper
Different error rate

L2 Error Norm over different error rate

- Wiberg L1
- Wiberg L2
- Rosper
Result for error free experiment

EXP 4:
Summary of Problem:
Number of frames = 10
Size of matrix: 20 x 30
Sampling rate = 0.40
Error Rate: 0.00
Error Amp: 1

Summary of Wiberg L1
Wiberg L1: Time elapsed = 0 hours : 0 minutes : 24 seconds
Wiberg L1: The residual through the 8 iterations are: 9.01
Wiberg L1: The L1 norm of error is 74.8112
Wiberg L1: The L2 norm of error is 4.9362
Wiberg L1: The L1 norm of sampled error is 9.0133
Wiberg L1: The L2 norm of sampled error is 1.3045
Wiberg L1: |detected E - E ground truth| = 9.0133
Wiberg L1: |recovered missing - w ground truth| = 65.7979

Summary of Wiberg L2
Wiberg L2: Time elapsed = 0 hours : 0 minutes : 2 seconds
Wiberg L2: The L1 norm of error is 0.0000
Wiberg L2: The L2 norm of error is 0.0000
Wiberg L2: The L1 norm of sampled error is 0.0000
Wiberg L2: The L2 norm of sampled error is 0.0000
Wiberg L2: |detected E - E ground truth| = 0.0000
Wiberg L2: |recovered missing - w ground truth| = 0.0000

Summary of Rosper
Rosper: Time elapsed = 0 hours : 0 minutes : 5 seconds
Rosper: The L1 norm of error is 121.1264
Rosper: The L2 norm of error is 10.3511
Rosper: The L1 norm of sampled error is 4.6059
Rosper: The L2 norm of sampled error is 0.4392
Rosper: |detected E - E ground truth| = 0.0316
Rosper: |recovered missing - E ground truth| = 116.5205
Experiment 7: 30% Large Outlier!
Large Sparse Outlier

Error L1 Norm under large outlier noise: 30% Uniform(-5,5)
Large Sparse Outlier

Error L2 Norm under large outlier noise: 30% Uniform(-5,5)
Experiment 9: Dense N(0,0.5)
Experiment 10: Dense $N(0,0.1)$
Dense Gaussian Noise

Error L1 Norm under Dense Gaussian Noise

- N(0,0.5)
- N(0,0.1)

- Wiberg L1
- Wiberg L2
- Rosper
Dense Gaussian Noise

Error L2 Norm under Dense Gaussian Noise

N(0,0.5)  N(0,0.1)

Series1  Series2  Series3
Experiment 11: Dense N(0,0.1)+0.2* Outlier Uniform(-5,5)
Experiment 12: Dense $N(0,0.1)+0.2\ast$ Outlier Uniform(-1,1)
Dense Gaussian Noise + Sparse Outlier

Error L1 Norm under dense $N(0,0.1)$ AND 20% sparse outlier

- Uni(-1,1)
- Uni(-5,5)

- Wiberg L1
- Wiberg L2
- Rosper
Dense Gaussian Noise + Sparse Outlier

Error L2 Norm under dense $\mathcal{N}(0,0.1)$ AND 20% sparse outlier

- Uni(-1,1)
- Uni(-5,5)

- Wiberg L1
- Wiberg L2
- Rosper
Diagonal band shaped data matrix
## Diagonal band shaped data matrix

<table>
<thead>
<tr>
<th>EXP1: 20% Uniform Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.45 Sample Rate</td>
</tr>
<tr>
<td>[m n r]=[20 66 4]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Wiberg L1</th>
<th>Rosper</th>
</tr>
</thead>
<tbody>
<tr>
<td># of iterations</td>
<td>17</td>
<td>2027</td>
</tr>
<tr>
<td>Running Time</td>
<td>6 min 36 sec</td>
<td>9 sec</td>
</tr>
<tr>
<td></td>
<td>Recovered E - E_ground_truth</td>
<td>106.72</td>
</tr>
<tr>
<td></td>
<td>Recovered Missing - W_ground_truth</td>
<td>253.35</td>
</tr>
<tr>
<td>L1 Norm:</td>
<td>Wgnd - Wr</td>
<td>304.35</td>
</tr>
<tr>
<td>L2 Norm:</td>
<td>Wgnd - Wr</td>
<td>16.38</td>
</tr>
<tr>
<td>L1 Norm:</td>
<td>mask(Wgnd - Wr)</td>
<td>51</td>
</tr>
<tr>
<td>L2 Norm:</td>
<td>mask(Wgnd - Wr)</td>
<td>4.55</td>
</tr>
</tbody>
</table>
Diagonal band shaped data matrix
## Banded Results table

<table>
<thead>
<tr>
<th>EXP2: 0 Uniform Error</th>
<th></th>
<th>Wiberg L1</th>
<th>Rosper</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.43 Sample Rate</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[m n r]=[20 56 4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># of iterations</td>
<td></td>
<td>35</td>
<td>5675</td>
</tr>
<tr>
<td>Running Time</td>
<td></td>
<td>6 min 59 sec</td>
<td>23 sec</td>
</tr>
<tr>
<td></td>
<td>Recovered E - E_ground_truth</td>
<td></td>
<td>&lt;0.01</td>
</tr>
<tr>
<td></td>
<td>Recovered Missing - W_ground_truth</td>
<td></td>
<td>61.6</td>
</tr>
<tr>
<td></td>
<td>L1 Norm:</td>
<td></td>
<td>61.61</td>
</tr>
<tr>
<td></td>
<td>L2 Norm:</td>
<td></td>
<td>6.22</td>
</tr>
<tr>
<td></td>
<td>L1 Norm:</td>
<td></td>
<td>&lt;0.01</td>
</tr>
<tr>
<td></td>
<td>L2 Norm:</td>
<td></td>
<td>&lt;0.01</td>
</tr>
</tbody>
</table>
Conclusion

• L1 norm is very good in terms of detecting outliers
• Wiberg L1 outperforms ALP and AQP in terms of smaller residual.
• The methods of alternatively optimizing multiple variables are likely to be faster than optimizing all variables together.
Conclusion

• Though claimed to be “efficient”, Wiberg L1 is slow and not scalable. Wiberg L2 however is a fast and reliable algorithm for many possible applications.

• Current state of factorization methods/robust matrix completion still not sufficient for robust application in most computer vision applications.
Conclusion

• Explicitly model noises in Rosper will have positive effects on results if the noise model is correct. Yet, it can be difficult tuning the weighting.

• In general, Rosper is similar to Wiberg L1 in performance, and is much faster.
Future works

• Deal with missing entries in rPCA?
• Generating the same 100 small random matrix and run the full test for Rosper and compare.
• Jacob, Martinez

• Spectrally optimal Factorization of Incomplete Matrix