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, Austrailia

Presented by

Wang Yuxiang

Department of Electrical and Computer Engineering,
National University of Singapore

NUS
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

NUS
National University of Singapore

Introduction: Problem formulation

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

$$\min_{U,V} ||\hat{W} \odot (Y - UV)||,$$

$Y \in \mathbb{R}^{m \times n}$ $U \in \mathbb{R}^{m \times r}$ $V \in \mathbb{R}^{r \times n}$ $\hat{W} \in \mathbb{R}^{m \times n}$

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

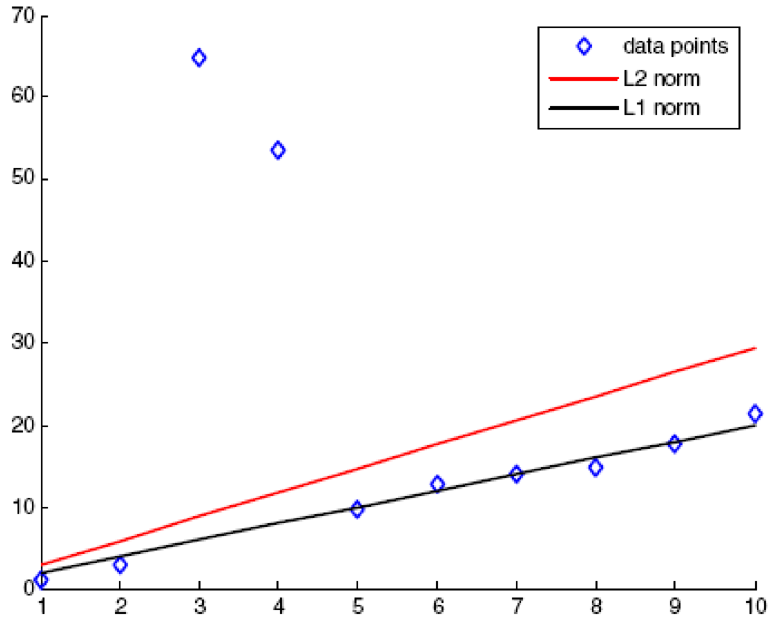


Figure 1. *Fit a line to 10 given data points. The two data points on upper-left are outliers.*

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 algoritm 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.

[6] Chandraker&Kriegman, certain combinatorial optimization method to reach global optimal.

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

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

$$\mathtt{U}^{(t)} = \arg\min_{\mathtt{U}} \|\mathtt{M} - \mathtt{U}\mathtt{V}^{(t)\top}\|_{L_1} \tag{8b}$$

Previous Works: Ke & Kanade

- Alternated Linear Programming:

$$E(\mathtt{V}) = \|\mathtt{M} - \mathtt{U}^{(t-1)}\mathtt{V}^\top\|_{L_1} = \sum_{j=1}^{n} \|\mathbf{m}_j - \mathtt{U}^{(t-1)}\mathbf{v}_j\|_1 \quad (9)$$

where $\mathbf{m}_j$ is the j-th column of $\mathtt{M}$, $\mathbf{v}_j$ is the j-th column of $\mathtt{V}^\top$. The problem of Eq. (8a) is therefore decomposed into *n independent small* sub-problems, each one optimizing $\mathbf{v}_j$:

$$\mathbf{v}_j = \arg\min_{\mathbf{x}} \|\mathtt{U}^{(t-1)}\mathbf{x} - \mathbf{m}_j\|_1 \quad (10)$$

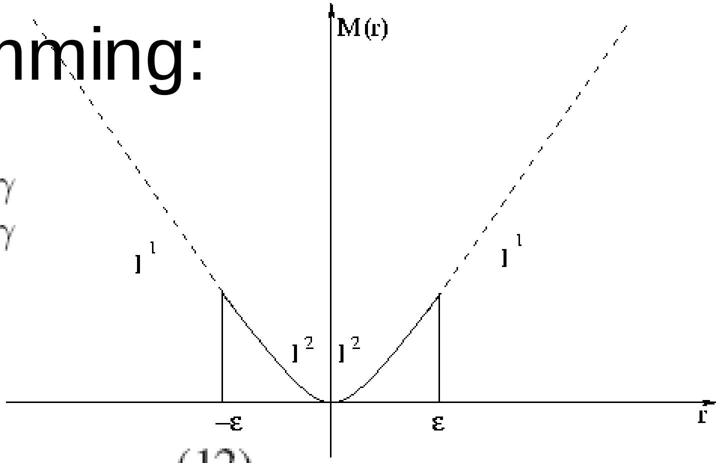
The *global* optimal solution of Eq. (10) can be found by the following linear program (LP):

$$\min_{\mathbf{x},\mathbf{t}} \mathbf{1}^\top\mathbf{t}$$

$$s.t. \; -\mathbf{t} \leq \mathtt{U}^{(t-1)}\mathbf{x} - \mathbf{m}_j \leq \mathbf{t} \quad (11)$$

Previous Works : Ke & Kanade

- Alternated Quadratic Programming:



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

$$\mathbf{v}_j = \arg\min_{\mathbf{x}} \rho(\mathrm{U}^{(t-1)}\mathbf{x} - \mathbf{m}_j) \qquad (12)$$

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]:

$$\min_{\mathbf{x},\mathbf{z},\mathbf{t}} \frac{1}{2}\|\mathbf{z}\|_2^2 + \gamma \mathbf{1}^\top \mathbf{t}$$
$$s.t. - \mathbf{t} \leq \mathrm{U}^{(t-1)}\mathbf{x} - \mathbf{m}_j - \mathbf{z} \leq \mathbf{t} \qquad (13)$$

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=diag(Ŵ), so Wy means the masked vec(Y).
- $\otimes$: Kronecker Product

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{bmatrix}.$$

Property of Kronecker Product

$$(\mathbf{B}^\top \otimes \mathbf{A})\,\mathrm{vec}(\mathbf{X}) = \mathrm{vec}(\mathbf{AXB}) = \mathrm{vec}(\mathbf{C}).$$

- Assume V is fixed, U is unknown, $\mathrm{vec}(UV)=\mathrm{vec}(IUV)=(V' \otimes I)\mathrm{vec}(U)$;
- Assume U is fixed and V is unknown, $\mathrm{vec}(UV)=\mathrm{vec}(UVI)=(I \otimes U)\mathrm{vec}(V)$;

Alternated least square (ALS)

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

$$\min_{v} ||Wy - W(I_n \otimes U)v||_2^2, \quad =||Wy\text{-}G(U)v||_2^2$$

$$\min_{u} ||Wy - W(V^T \otimes I_m)u||_2^2, \quad = ||Wy\text{-}F(V)u||_2^2$$

$f = Fu - Wy = Gv - Wy;$

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

Take partial derivative of Φ and assign to 0, we get:

$u = (F^T F)^{-1} F^T (Wy); \ v = (G^T G)^{-1} G^T (Wy);$

Gauss-Newton Method

- Let $x=[u^T\ v^T]$, $\Phi(U,V) = 1/2f^Tf$
- Newton's method: To attain $\partial\Phi/\partial x=0$ using the step δ, satisfying:

$$\partial^2\Phi/\partial x^2\ \Delta x + \partial\Phi/\partial x = 0 \qquad (*)$$

- Calculate the first and second derivative:

$$\partial\Phi/\partial x = f^T\partial f/\partial x$$

$$\partial^2\Phi/\partial x^2 = (\partial f/\partial x)^T\partial f/\partial x + f^T\ \partial^2 f/\partial x^2 \approx (\partial f/\partial x)^T\partial f/\partial x$$

- Note that $\partial f/\partial x = [\partial f/\partial u\ \partial f/\partial v] = [F\ G]$

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:

$$\partial^2 \psi / \partial u^2 \, \Delta u + \partial \psi / \partial u = 0$$

We need to know the first and second derivative of ψ w.r.t. u.

Wiberg Algorithm

- Again, using Gauss-Newton Approx, we have:

  $\partial \psi / \partial u = g^T \partial g / \partial u$

  $\partial^2 \psi / \partial u^2 = (\partial g/\partial u)^T \partial g/\partial u + g^T \partial^2 g/\partial u^2 \approx (\partial g/\partial u)^T \partial g/\partial u$

- Using chain rule:

  $\partial g/\partial u = \partial f/\partial u + \partial f/\partial v \, \partial v/\partial u = F+G(\partial v/\partial u)$

- At optimal (u,v): $\partial \Phi/\partial u = 0$ AND $\partial \Phi/\partial v = 0$ (KKT)

  $\partial \Phi/\partial u = \partial \psi / \partial u$ will be enforce to 0 by Gauss-Newton.

  $\partial \Phi/\partial v=0$ is irrelevant to u, so:

  $\partial^2 \Phi/(\partial v \partial u)= \partial (f^T \partial f/\partial v)/ \partial u = \partial (f^T G)/ \partial u$

  $= (\partial f/\partial v * \partial v/\partial u+ \partial f/\partial u)^T G + f^T (\partial G)/ \partial u \approx 0$

Wiberg Algorithm

- $(\partial f/\partial v * \partial v/\partial u + \partial f/\partial u)^T G = 0$

  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 Δu for Gauss-Newton update in:

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

  This is a least square problem. As derived in Okatani Paper [16].

Insufficient Rank of $Q_G F$

- Okatani proved that this $Q_G F$ 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 Δu.
- Yet, in this Eriksson paper, it is not mentioned. (corresponds to the Insufficient Rank of Jacobian)

Connection to Linearized Model

Consider

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

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 $\partial g/\partial u = Q_G F$

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

Since Wy is const. so $\partial g'/\partial u = \partial g/\partial u$

Connection to Linearized Model

- Now it becomes

  Minimize $\| -Wy + g'(u_k) + \partial g'/\partial u \, \Delta u \|$

- This corresponds to Equation (8) in the paper and $J_k = \partial g'/\partial u|_{u=u_k} = Q_G F$.

  (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

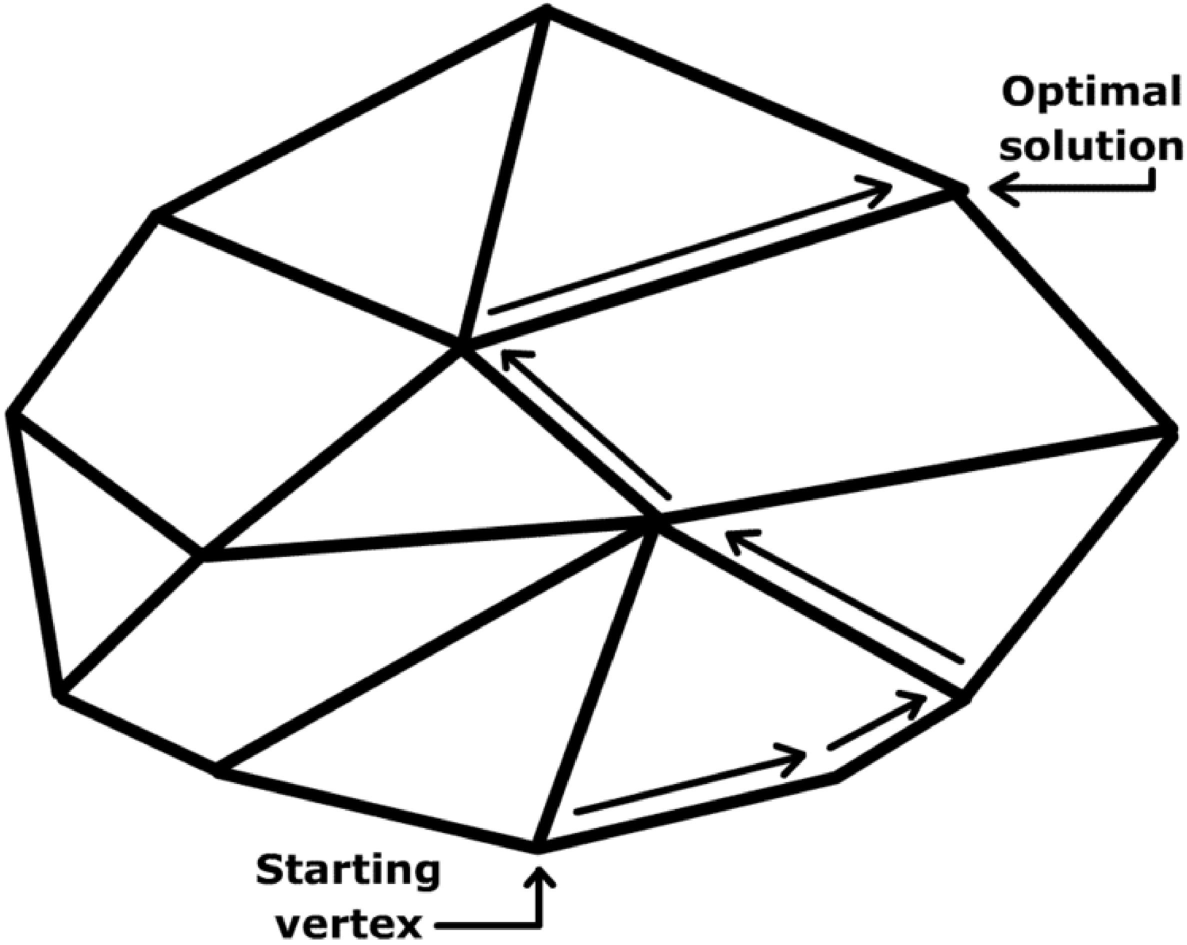
$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & c^T x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

- 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*m non-singular.
- $x=[x_B \ x_N]$ and $x_N = 0$, then x is b.s.
- If $x_B >= 0$, 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:

$$\frac{\partial x_B^*}{\partial B} = -(x_B^*)^T \otimes B^{-1} \quad (13)$$

$$\frac{\partial x_B^*}{\partial N} = 0 \quad (14)$$

$$\frac{\partial x_B^*}{\partial b} = B^{-1} \quad (15)$$

$$\frac{\partial x_N^*}{\partial A} = \frac{\partial x_N^*}{\partial b} = 0. \quad (16)$$

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 δ of U using LP again.

3. Then update U and proceed in a gradient descent manner.

Proposed Algorithm: Wiberg L1

$$v^*(U) = \arg\min_v ||Wy - W(I_n \otimes U)v||_1,$$

$$u^*(V) = \arg\min_u ||Wy - W(V^T \otimes I_m)u||_1,$$

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

$$\min_U \; f(U) = ||Wy - WUV^*(U)||_1 =$$

$$= ||Wy - \phi_1(U)||_1.$$

- Φ_1(u) = vec(WUV*(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.)

$$\min_{v^+, v^-, t, s} \quad \begin{bmatrix} 0 & 0 & 1^T & 0 \end{bmatrix} \begin{bmatrix} v^+ \\ v^- \\ t \\ s \end{bmatrix} \tag{33}$$

$$\text{s.t.} \quad \underbrace{\begin{bmatrix} -G(U) & G(U) & -I & I \\ G(U) & -G(U) & -I & \end{bmatrix}}_{A(U)} \begin{bmatrix} v^+ \\ v^- \\ t \\ s \end{bmatrix} = \underbrace{\begin{bmatrix} -Wy \\ Wy \end{bmatrix}}_{b} \tag{34}$$

$$v^+, v^-, t, s \geq 0 \tag{35}$$

$$v^+, v^- \in \mathbb{R}^{rn}, \;\; t \in \mathbb{R}^{mn}, \;\; s \in \mathbb{R}^{2mn}. \tag{36}$$

Given v*, linearize the model and find the optimal Δu

- In order to do that, we must calculate J(U)

  $\partial\Phi_1/\partial u = \partial\Phi_1/\partial u + \partial\Phi_1/\partial v^* \partial v/\partial u$

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

- 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 Δu

$$\frac{\partial z^*}{\partial U} = \begin{bmatrix} \frac{\partial v^{*+}}{\partial U} \\ \frac{\partial v^{*-}}{\partial U} \\ \frac{\partial t^*}{\partial U} \\ \frac{\partial s^*}{\partial U} \end{bmatrix} = \frac{\partial z^*}{\partial B}\frac{\partial B}{\partial G}\frac{\partial G}{\partial U}$$

$$\frac{\partial G}{\partial U} = (I_{nr} \otimes W)(I_n \otimes T_{r,n} \otimes I_m)(\mathrm{vec}(I_n) \otimes I_{mr}) \quad (37)$$

$$\begin{aligned} \frac{\partial B}{\partial G} &= \frac{\partial (AQ)}{\partial G} = (Q^T \otimes I_{2mn})\frac{\partial A}{\partial G} = \\ &= (Q^T \otimes I_{2mn})\left[\begin{array}{cc} \frac{\partial}{\partial G}\left(\begin{smallmatrix} -G & G \\ G & -G \end{smallmatrix}\right) & 0 \end{array}\right] = \\ &= (Q^T \otimes I_{2mn})\left[\begin{array}{cc} \frac{\partial\left(\left[\begin{smallmatrix} -1 & 1 \\ 1 & -1 \end{smallmatrix}\right] \otimes G\right)}{\partial G} & 0 \end{array}\right] \end{aligned} \quad (38)$$

$$\frac{\partial z^*}{\partial B} = Q\frac{\partial z_B^*}{\partial B} = -Q\left((z_B^*)^T \otimes B^{-1}\right) \quad (39)$$

Very high dimension even for small data matrix.
Even sparse representation doesn't work.

Minimizing L1 norm over Δu

- To find the optimal Δu, 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:

$$\text{min. } ||Wy - \Phi_1(u_k) - J(u_k)(\Delta u)||$$

LP to find Δu and minimize L1 norm

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

$$\min_{\delta,t} \quad \begin{bmatrix} 0 & 1^T \end{bmatrix} \begin{bmatrix} \delta \\ t \end{bmatrix} \tag{43}$$

$$\text{s.t.} \quad \begin{bmatrix} -J(U)-I \\ J(U)-I \end{bmatrix} \begin{bmatrix} \delta \\ t \end{bmatrix} = \begin{bmatrix} -(Wy-W\text{vec}(UV^*)) \\ Wy-W\text{vec}(UV^*) \end{bmatrix} \tag{44}$$

$$||\delta||_1 \leq \mu \tag{45}$$

$$\delta \in \mathbb{R}^{mr}, \quad t \in \mathbb{R}^{mn}. \tag{46}$$

- Again, this is solved by Simplex method.

Wiberg L1 Summary

Algorithm 1: L_1-Wiberg Algorithm

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

$k = 0$;

repeat

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

    Solve the subproblem (43)-(46) to obtain δ_k^* ;

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

    if $gain \geq \epsilon$ **then**

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

    end

    if $gain < \eta_1$ **then**

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

    end

    if $gain > \eta_2$ **then**

        $\mu = c\mu$

    end

    $k = k + 1$;

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 Uni(-5,5)
- Tested over 100 data matrices to obtain a statistic.

Histogram: Frequency vs. Error

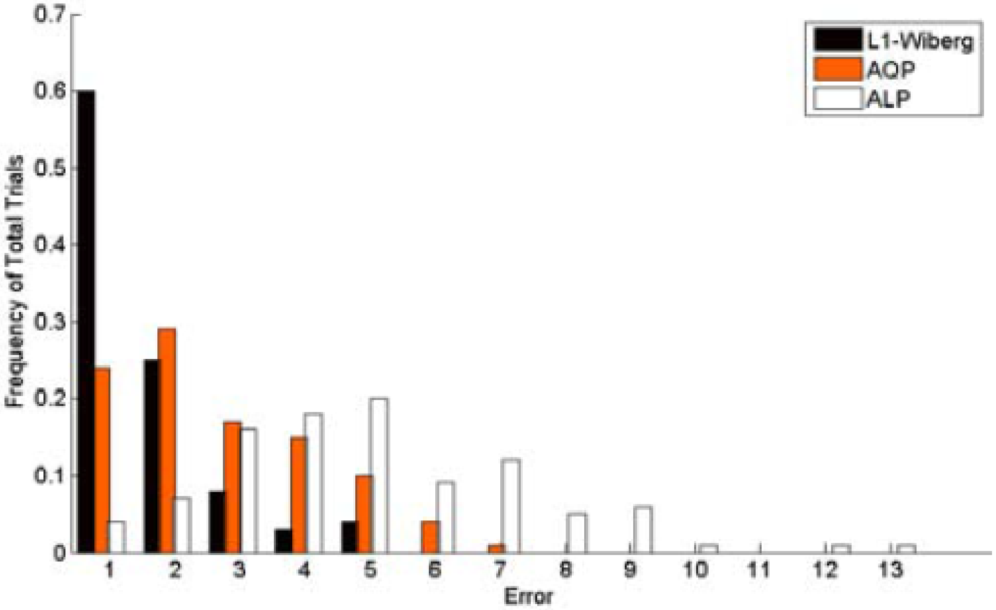


Figure 1. A histogram representing the frequency of different magnitudes of error in the estimate generated by each of the methods. [Frequency vs. Error]

Speed of convergence: Residual vs. Iteration

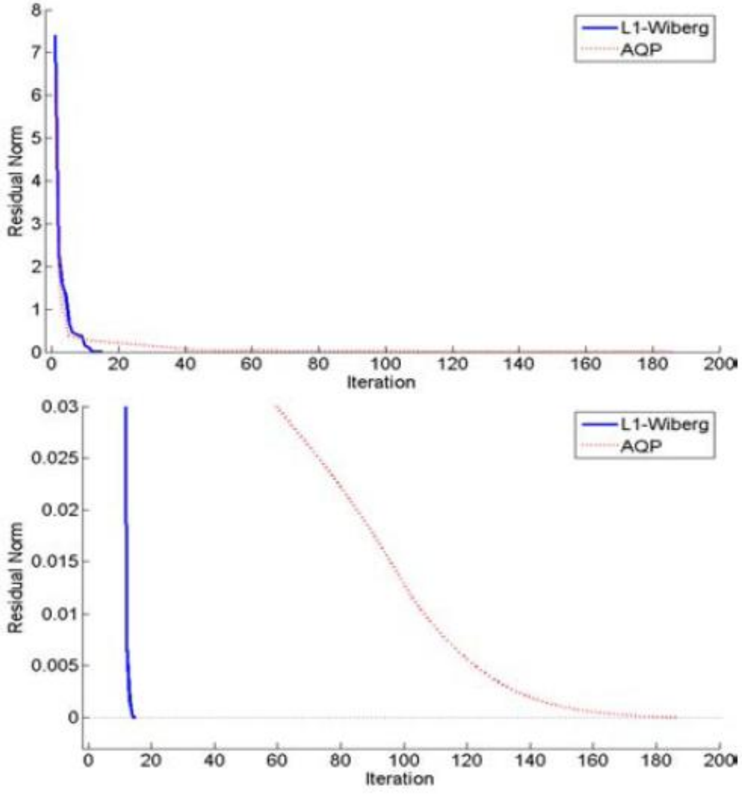


Figure 2. Plots showing the norm of the residual at each iteration of two randomly generated tests for both the L_1 Wiberg and alternated quadratic programming algorithms. [Residual norm vs. Iteration]

Speed of convergence: log error vs. Iteration

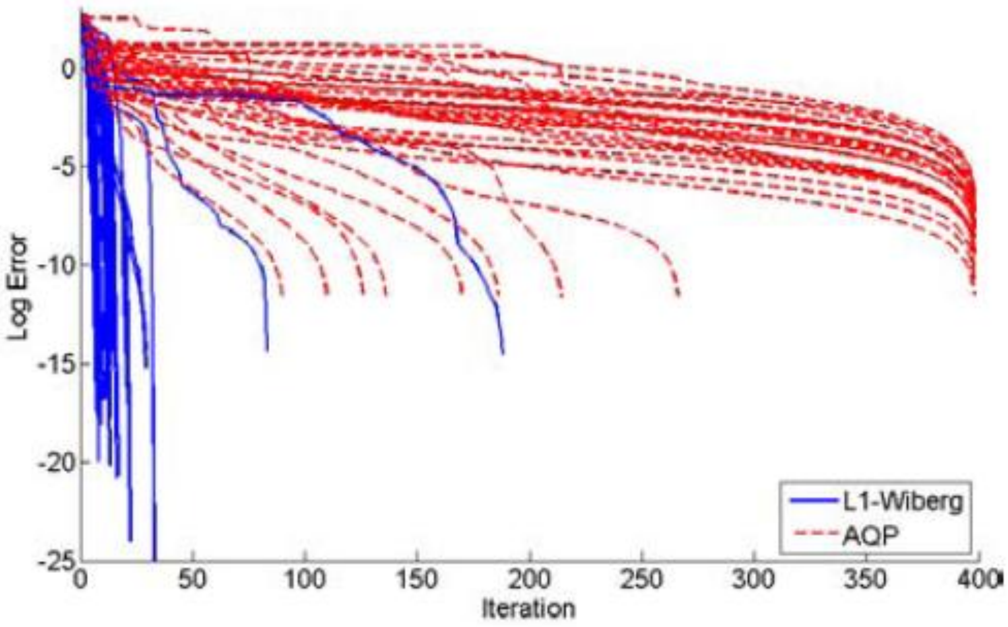


Figure 3. A plot showing the convergence rate of the alternated quadratic programming and L_1-Wiberg algorithms over 100 trials. The error are presented on a logarithmic scale. [Log Error vs. Iteration]

Table of Results

Algorithm	Alt. LP [13]	Alt. QP [13]	Wiberg $L_1(Alg.1)$
Error (L_1)	4.60	2.29	**1.01**
Execution Time (sec)	0.16	93.57	1.51
# Iterations	4.72	177.64*	21.77
# LP/QP solved	9.44	355.28*	24.13
Time per LP/QP	0.016	0.264*	0.061

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

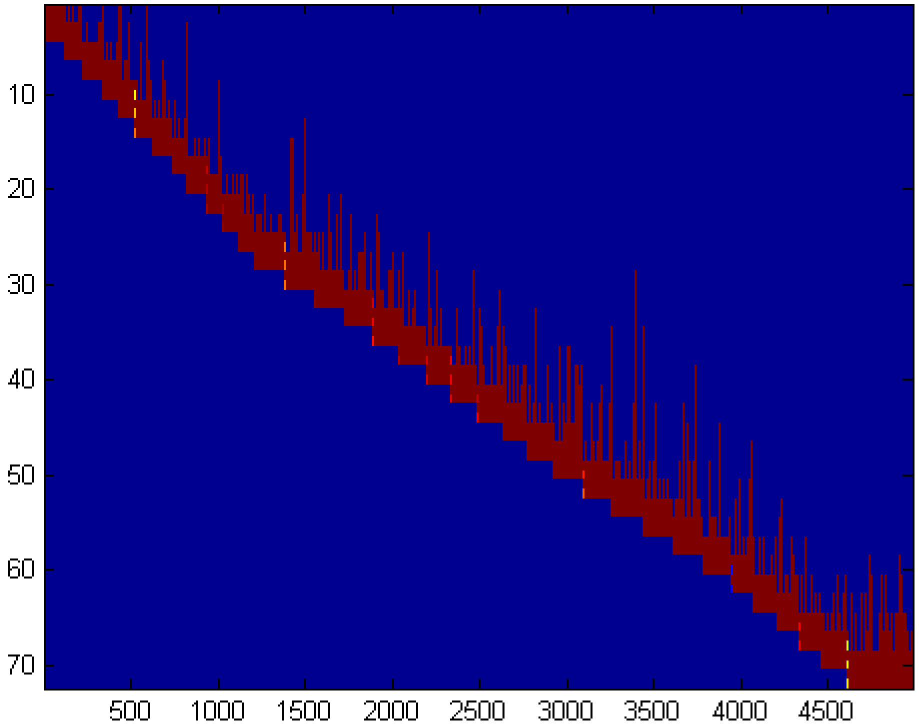
Table 1. *The averaged results from running* 100 *synthetic experiments.*

- 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

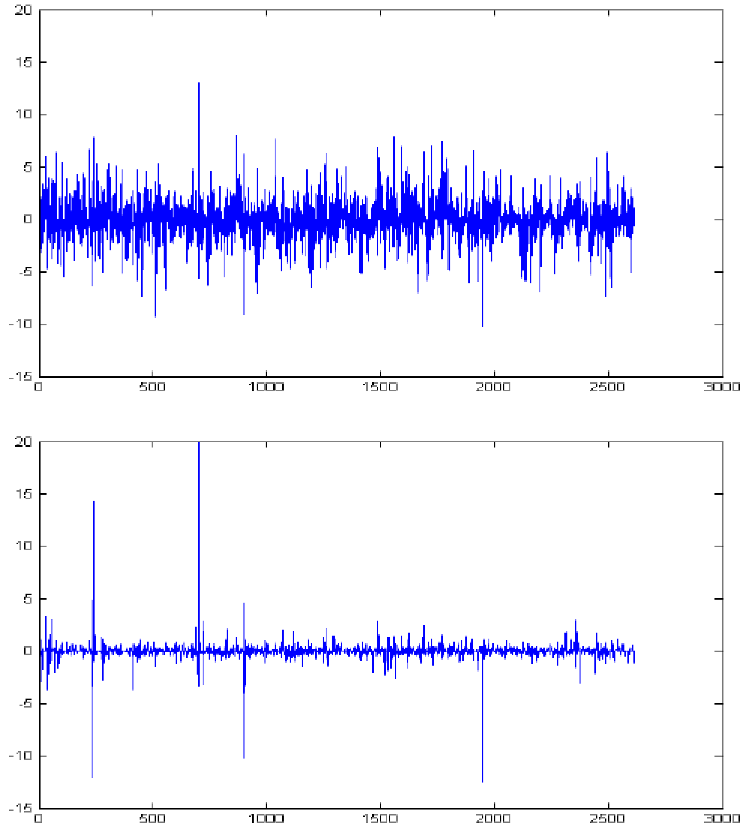


Figure 4. Resulting residuals using the standard Wiberg algorithm (top), and our proposed L_1-Wiberg algorithm (bottom).

Result table and Reconstructed point cloud

Algorithm	Alt. QP [13]	Wiberg L_2 [20]	Wiberg L_1 (Alg. 1)
RMS Error of Inliers	-	2.029	0.862
Execution Time	>4 hrs	3 min 2 sec	17 min 44 sec

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

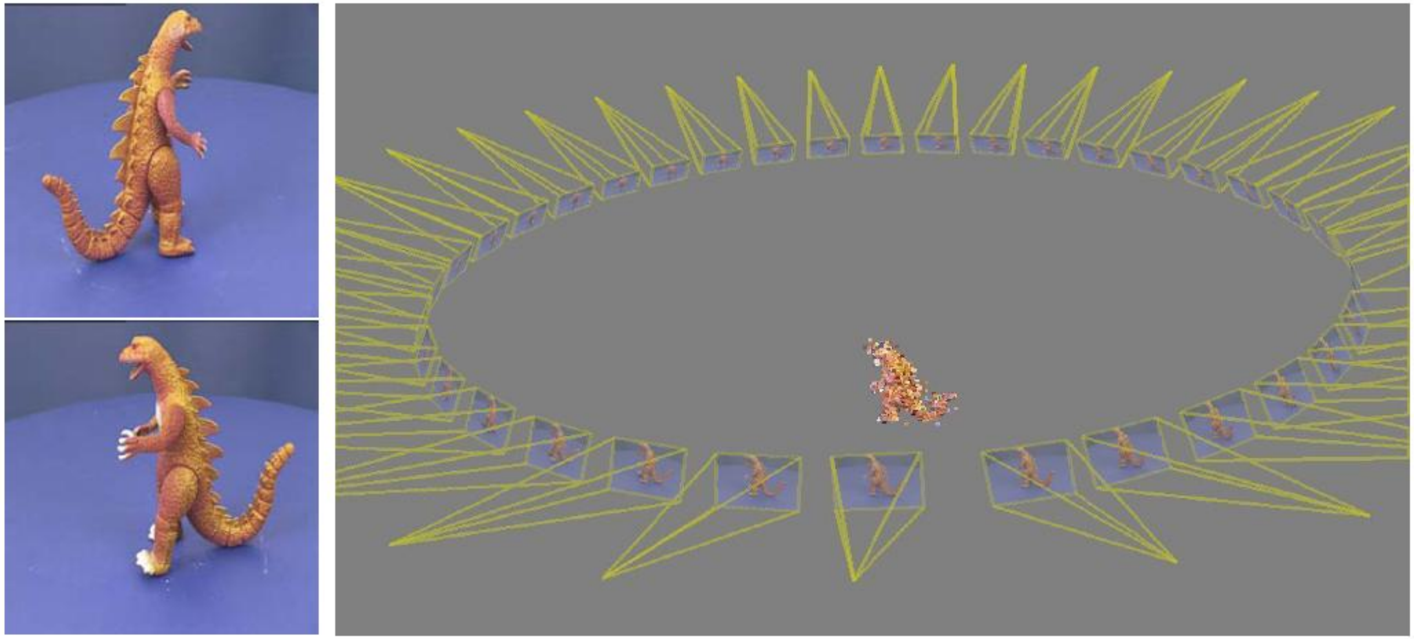


Figure 5. *Images from the dinosaur sequence, and the resulting reconstruction using our proposed method.*

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

$$\min \frac{1}{2}\|\mathcal{A}(W) + \mathcal{A}(E) - \mathcal{A}(\widehat{W})\|_F^2 + \mu\|E\|_{2,1}$$
$$s.t.\ rank(W) \leqslant r$$

- 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. Slow and not scalable. (Not working for matrices as small as 20*80)
- **Original Wiberg (L2):** provided by Okatani, citation [16]. Features checking if a data matrix is well-posed for the algorithm. (If $Q_F G$ 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.

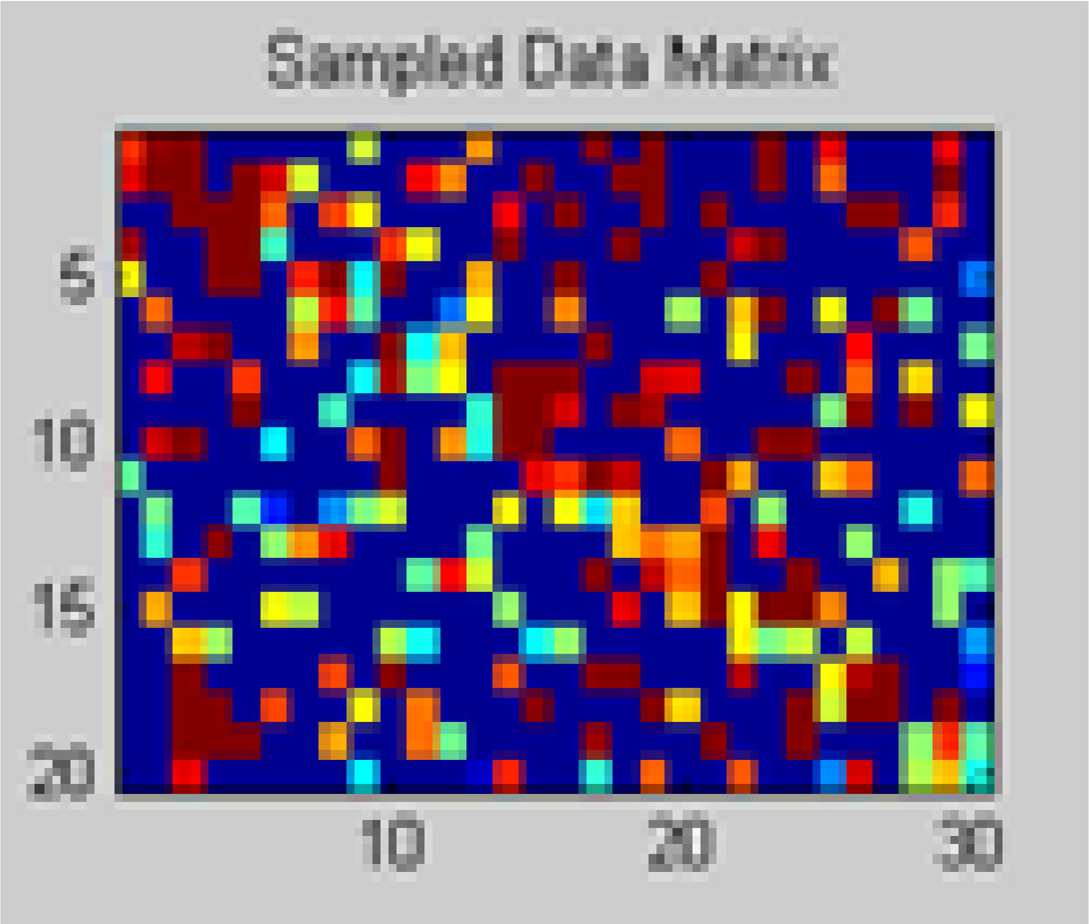
Difference in Error Model

- 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.

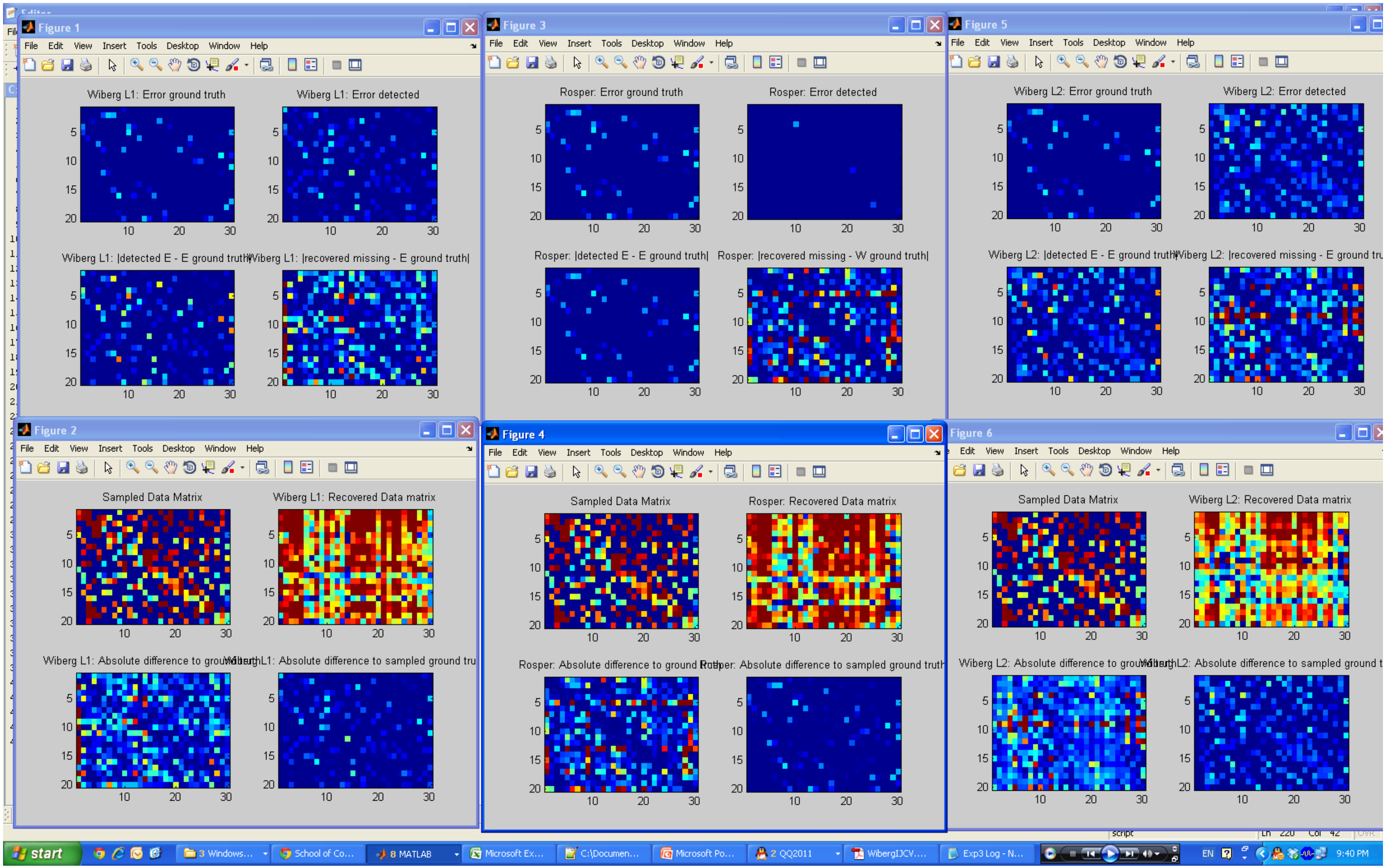
Experiment design

- 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

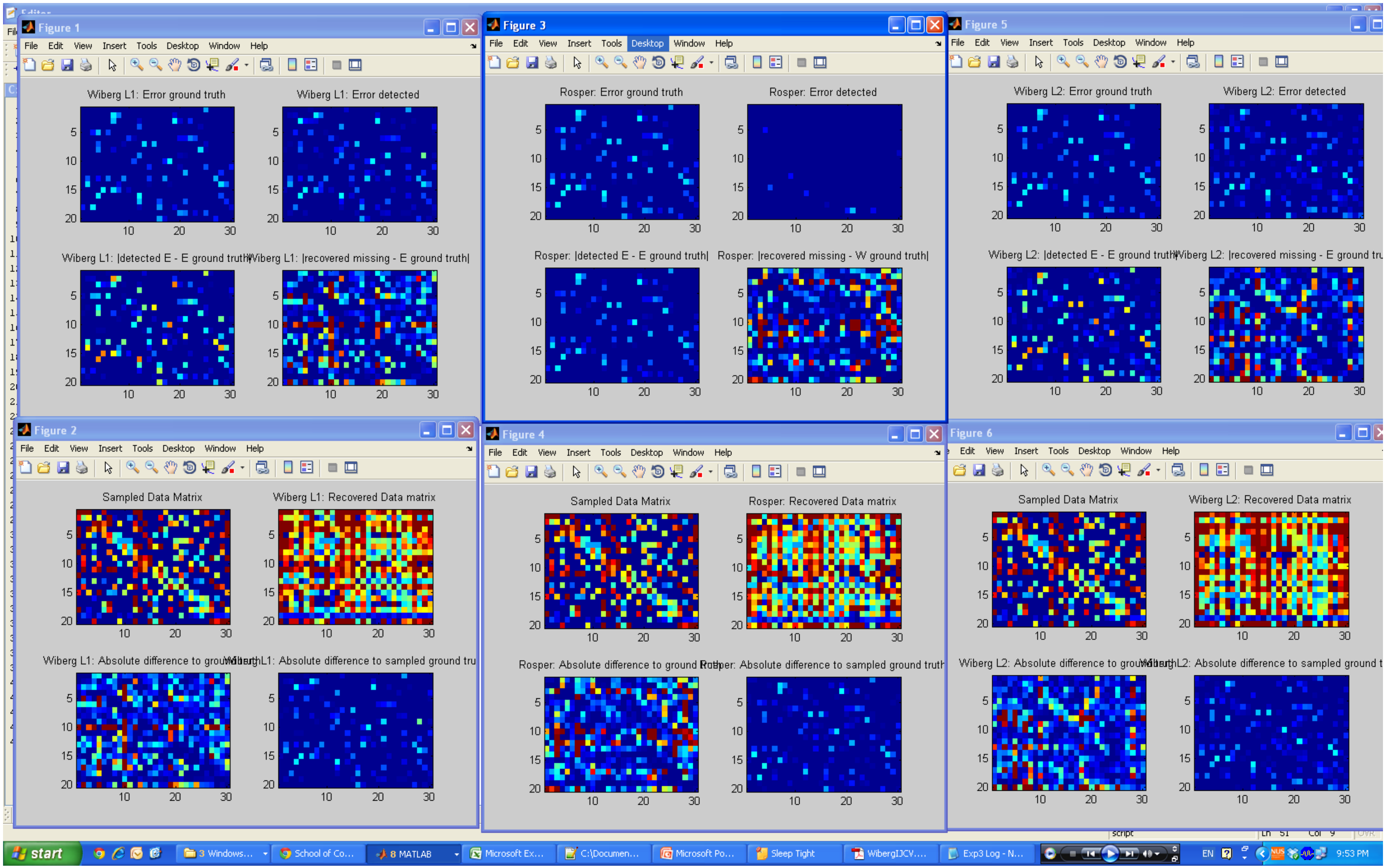
An illustration of the sampled data matrix



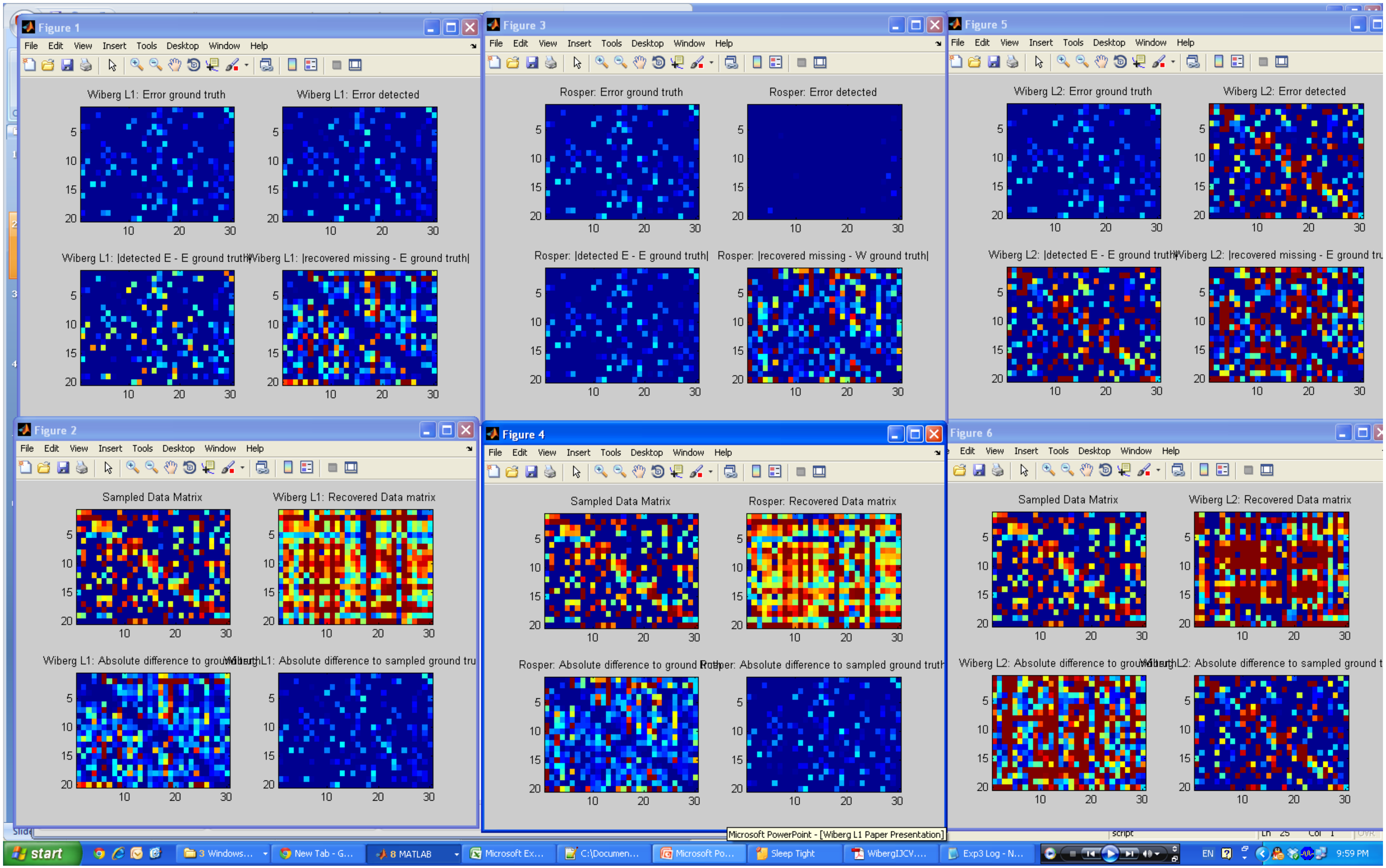
Experiment 3: 0.3 Error



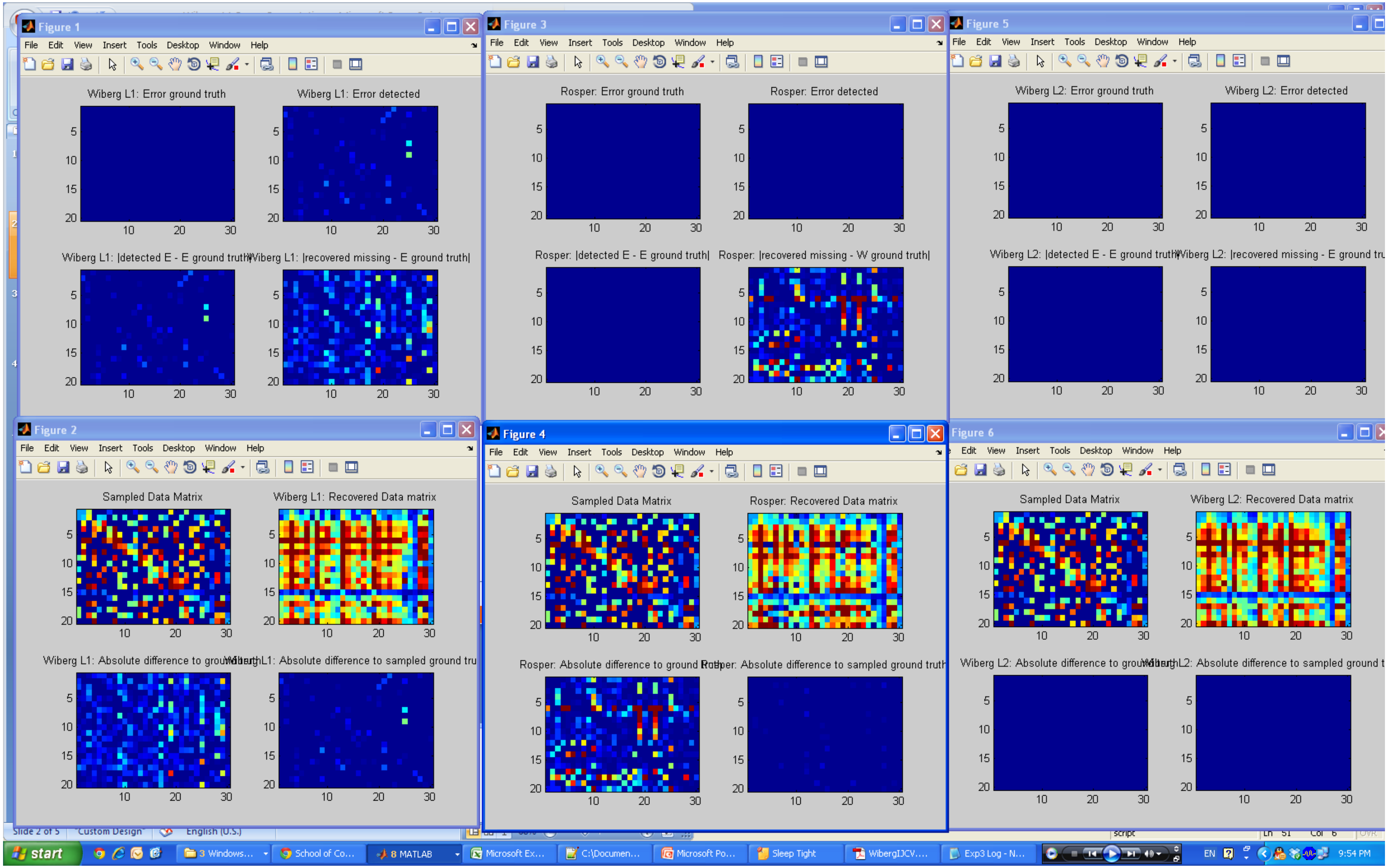
Experiment 5: 0.5 Error



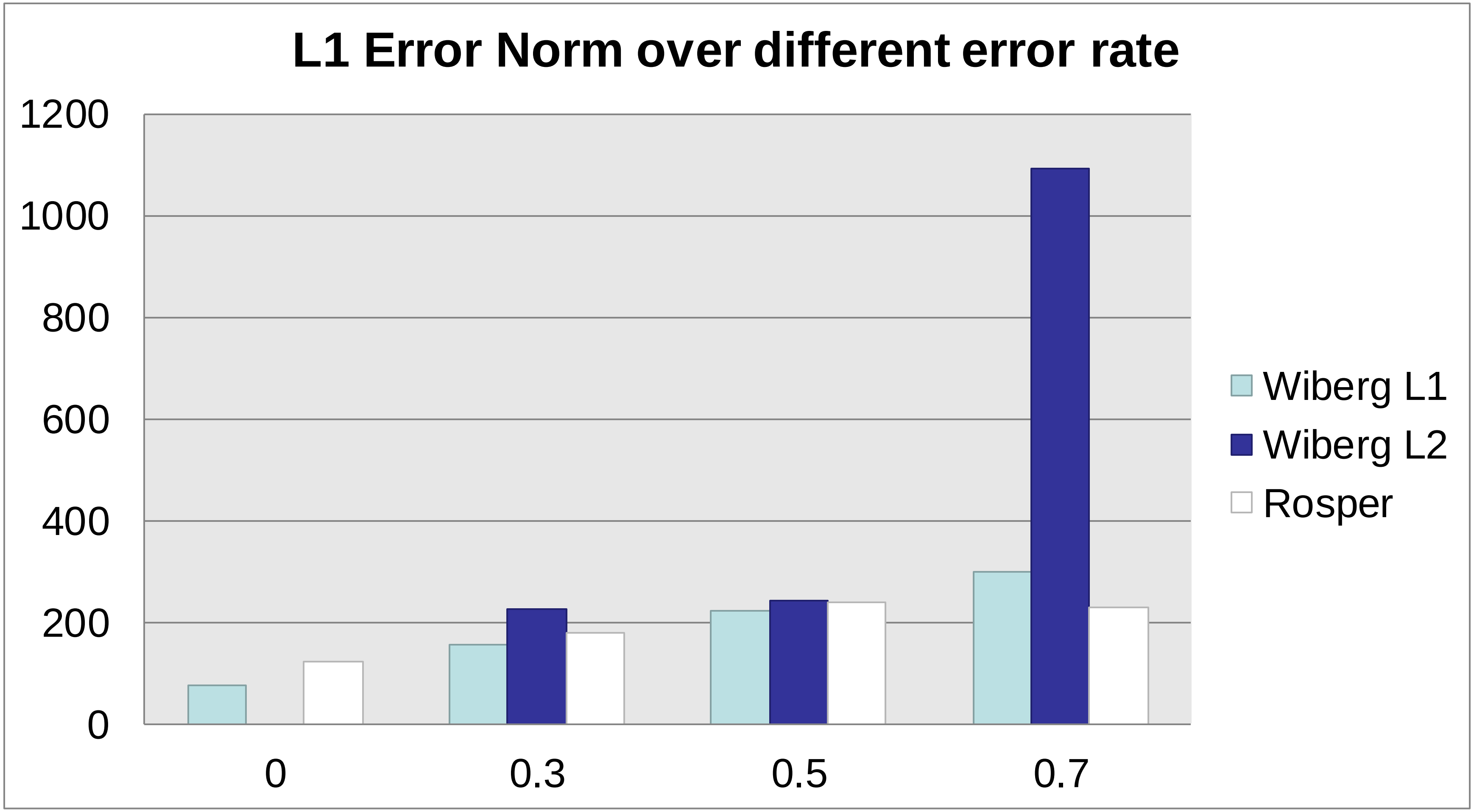
Experiment 6: 0.7 Error



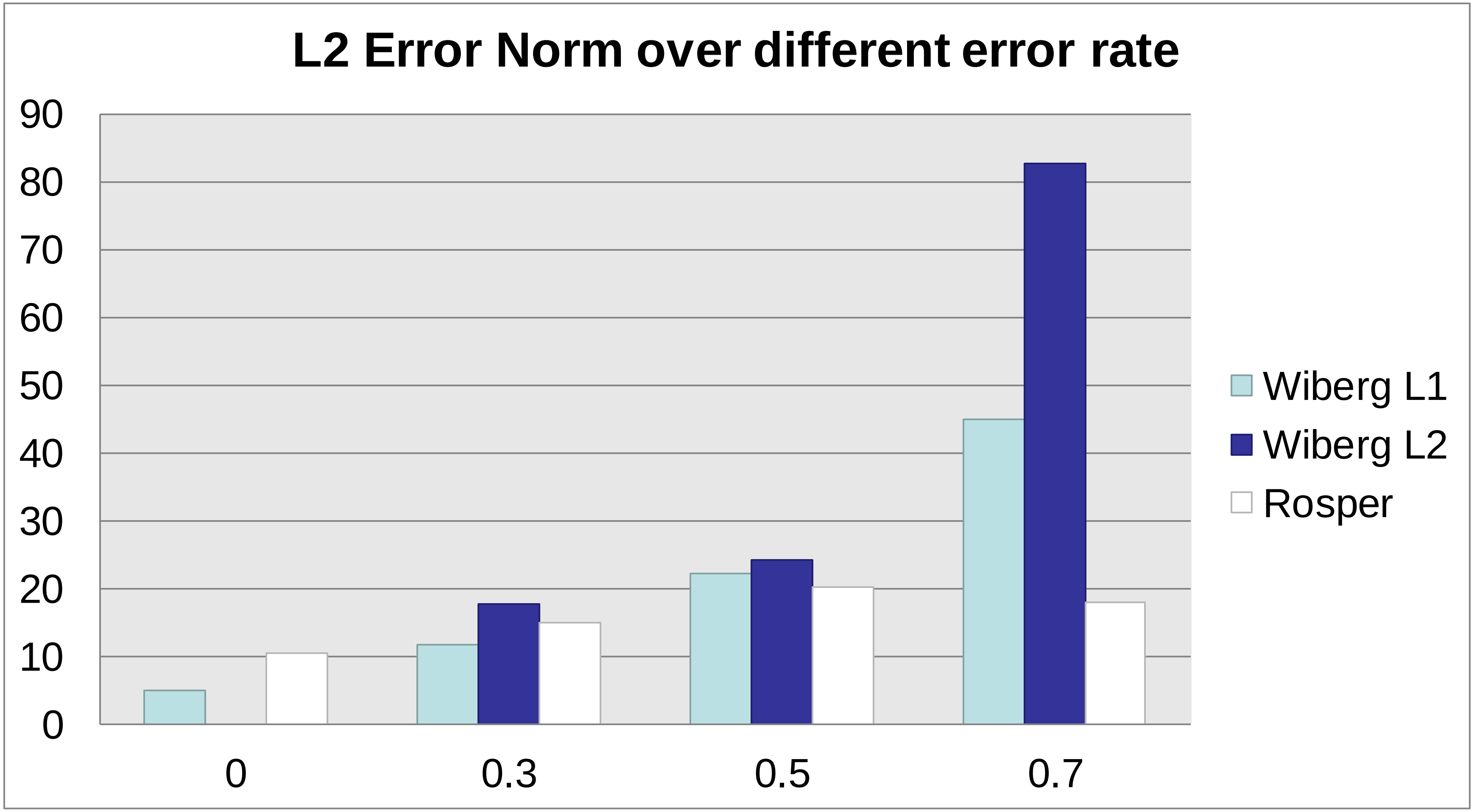
Experiment 4: 0 Error



Different error rate



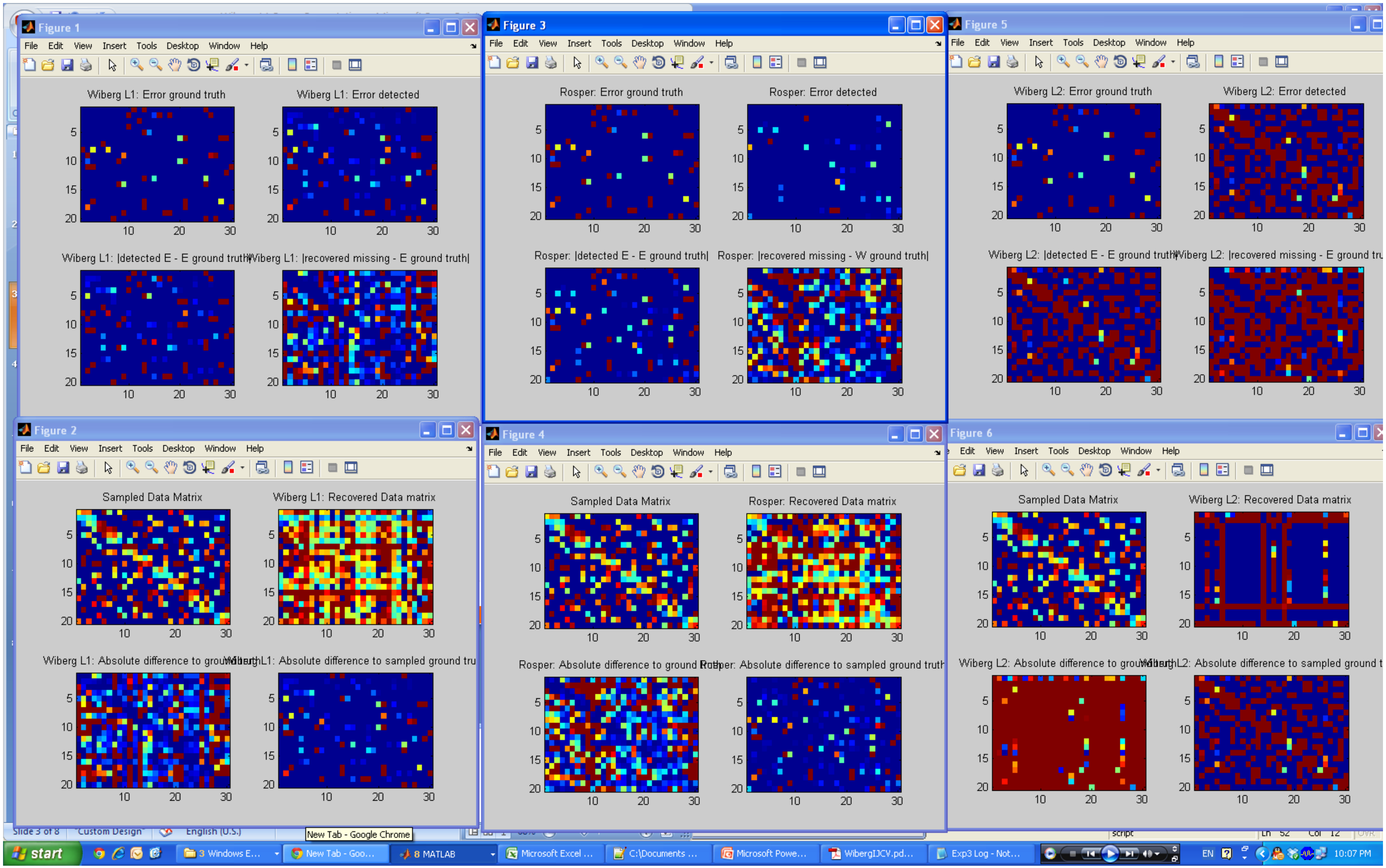
Different error rate



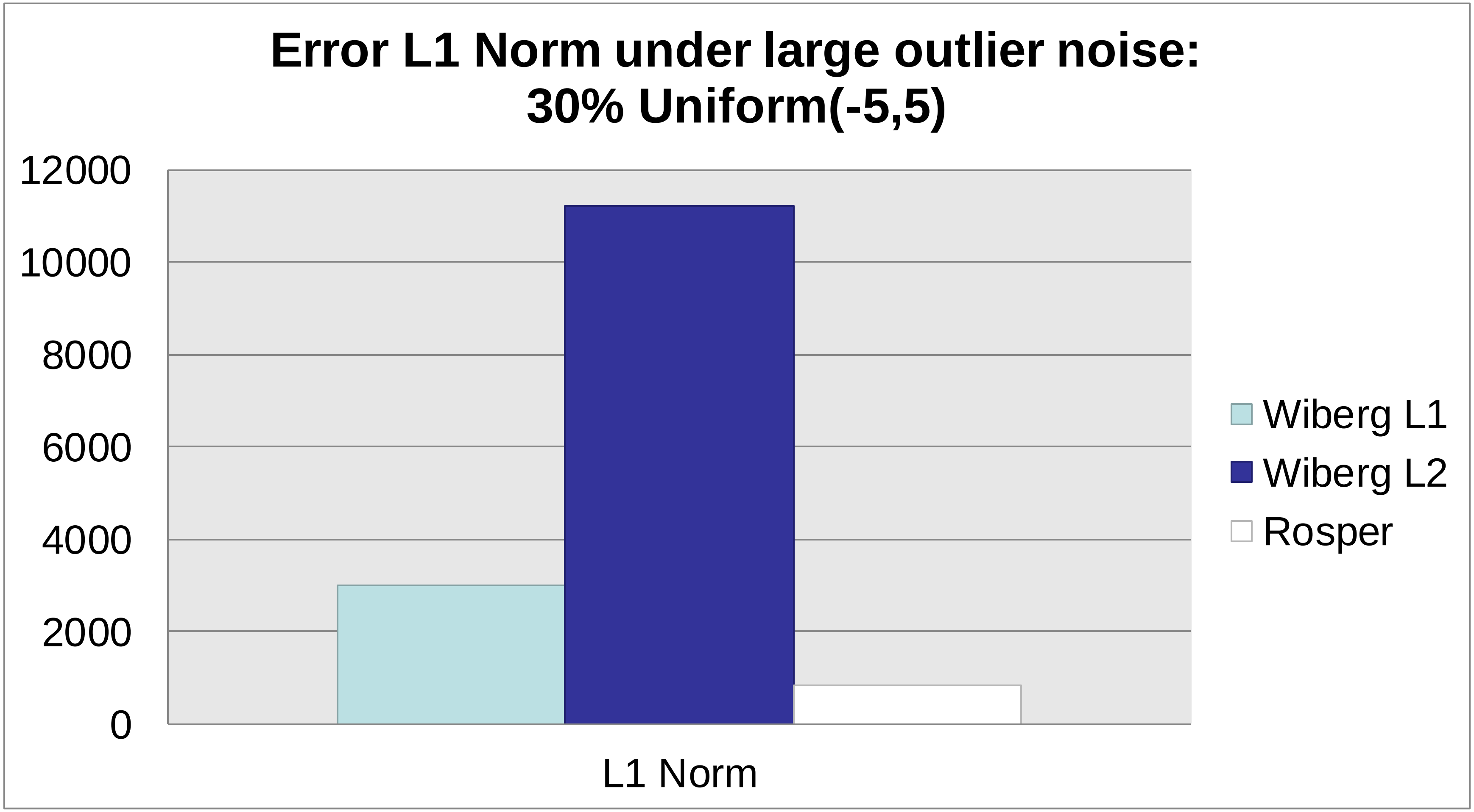
Result for error free experiment

```
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
```

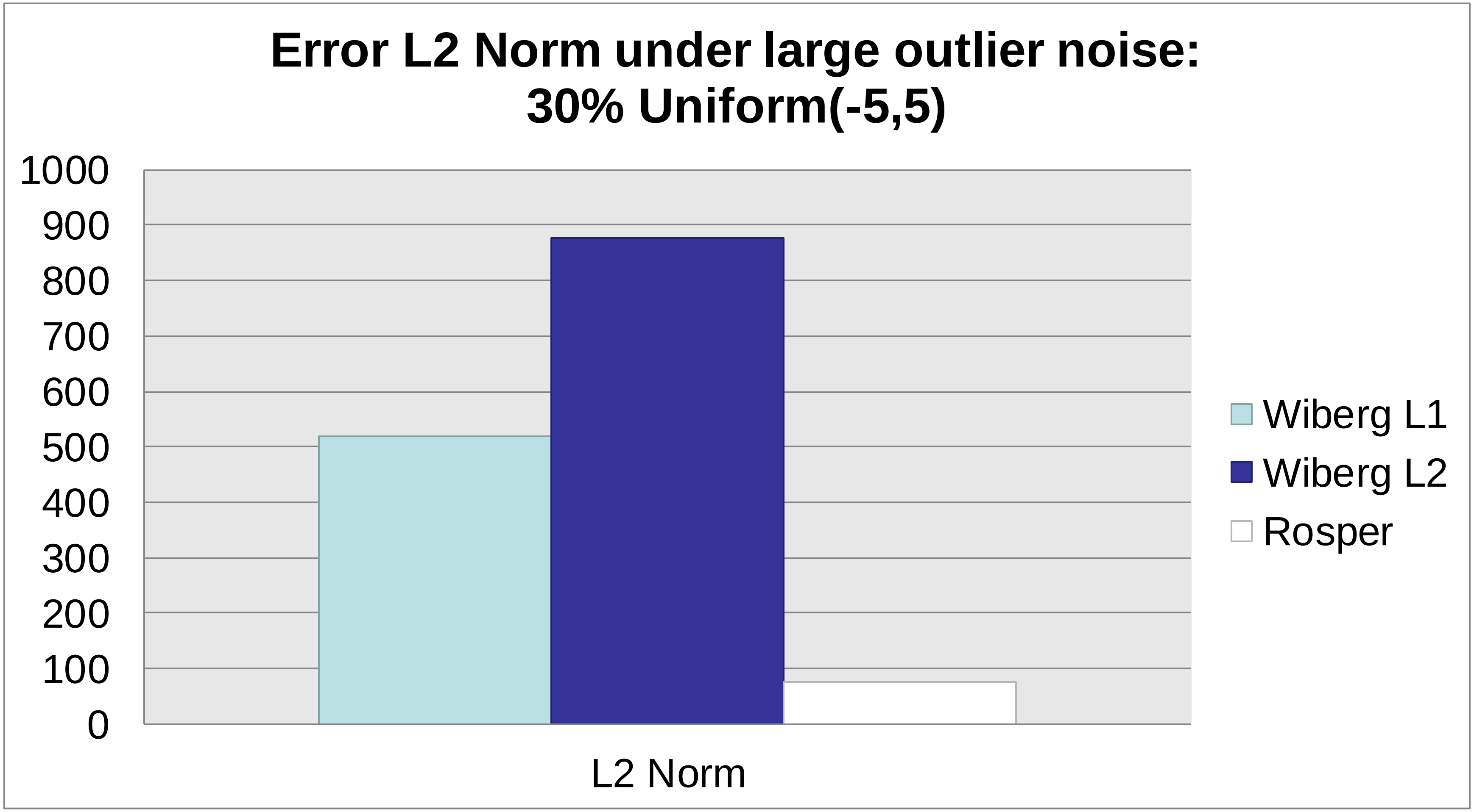
Experiment 7: 30% Large Outlier!



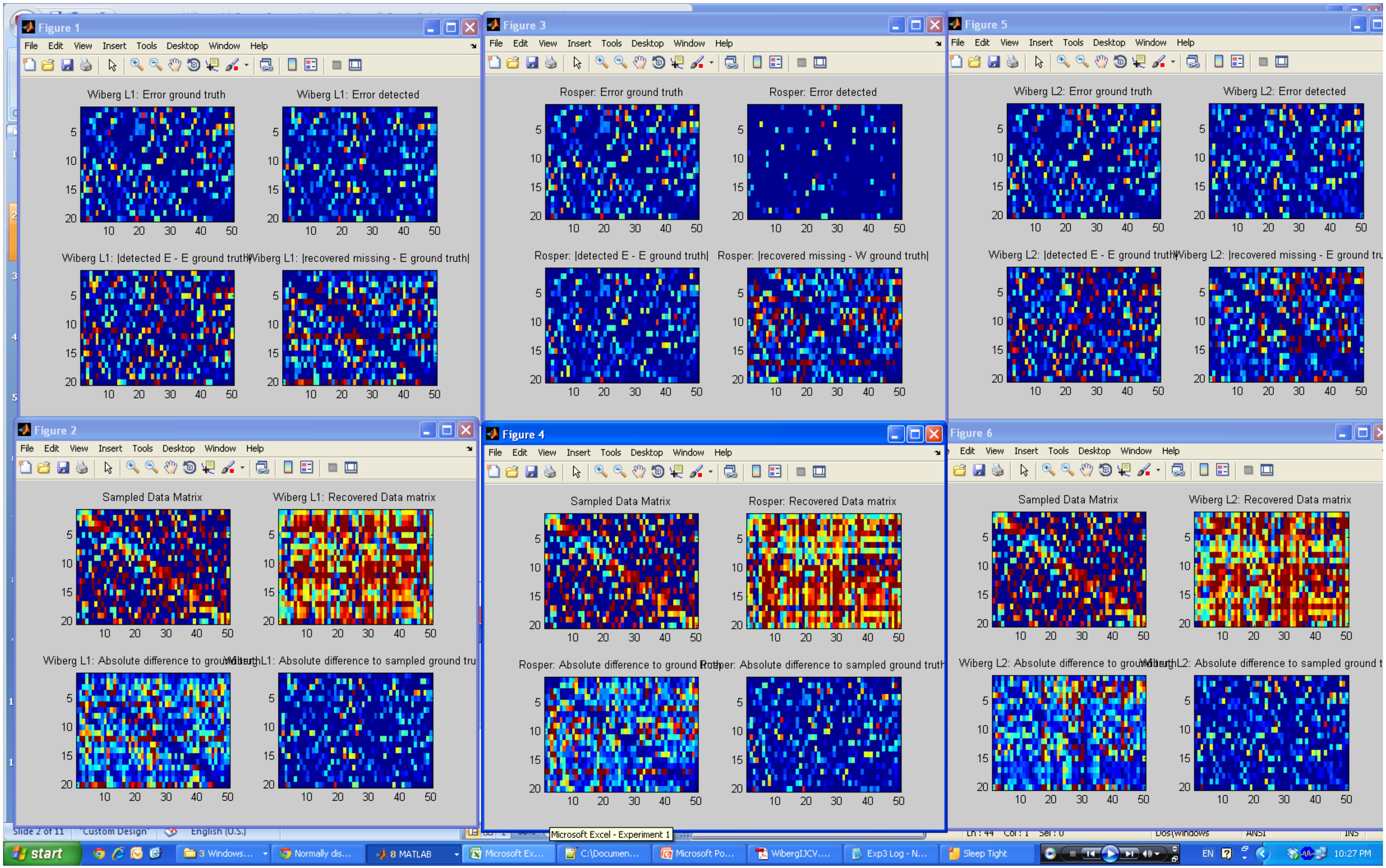
Large Sparse Outlier



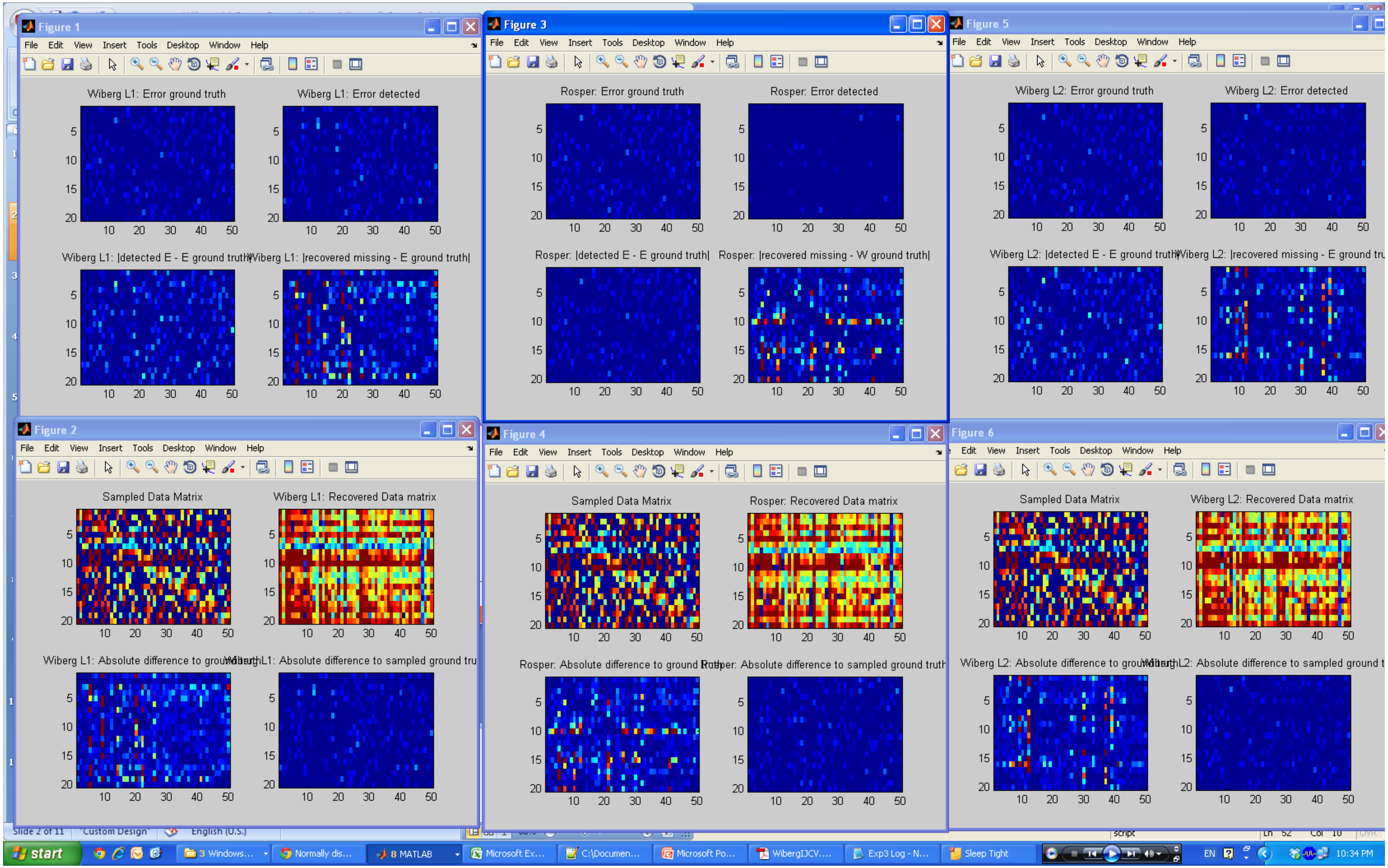
Large Sparse Outlier



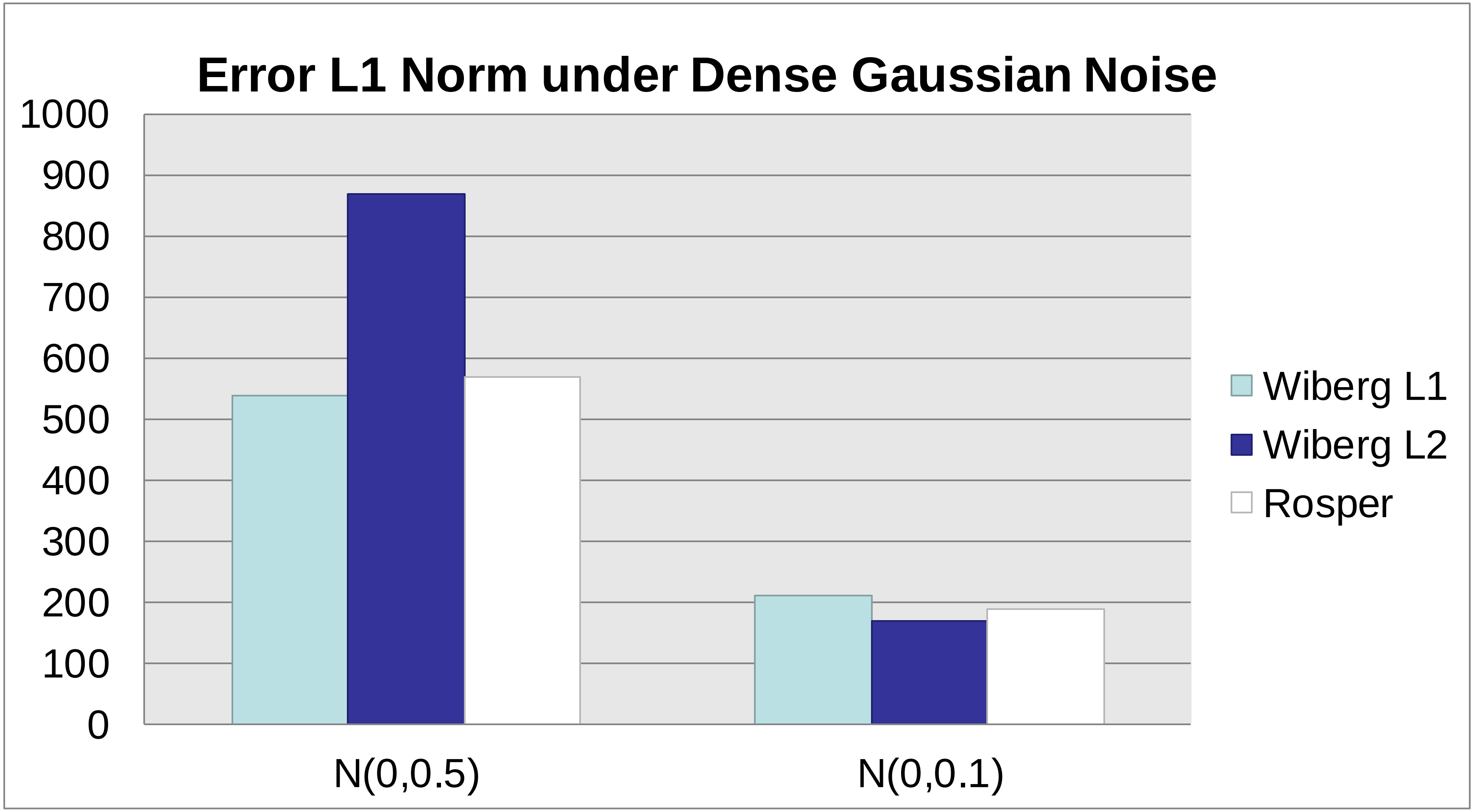
Experiment 9:Dense N(0,0.5)



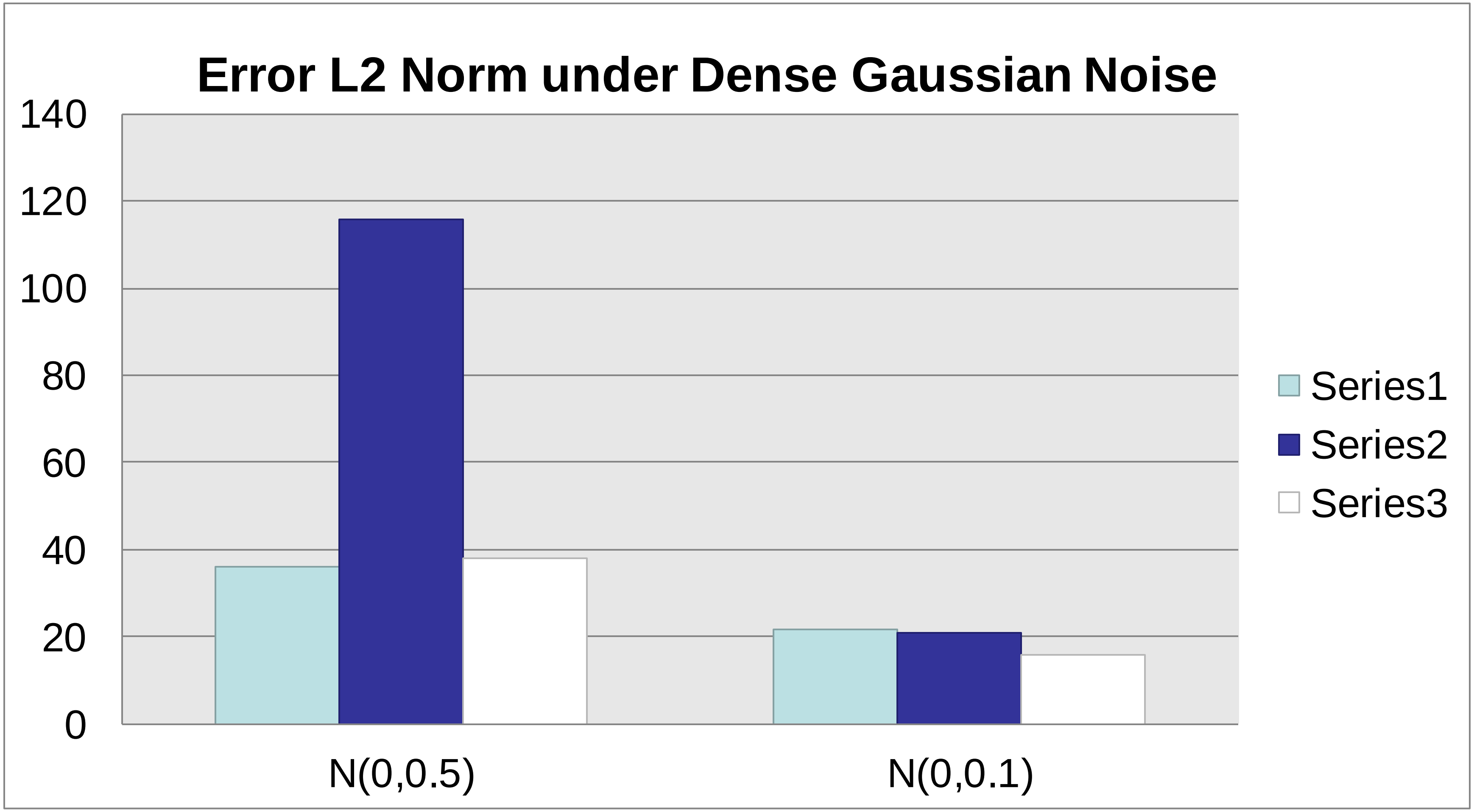
Experiment 10:Dense N(0,0.1)



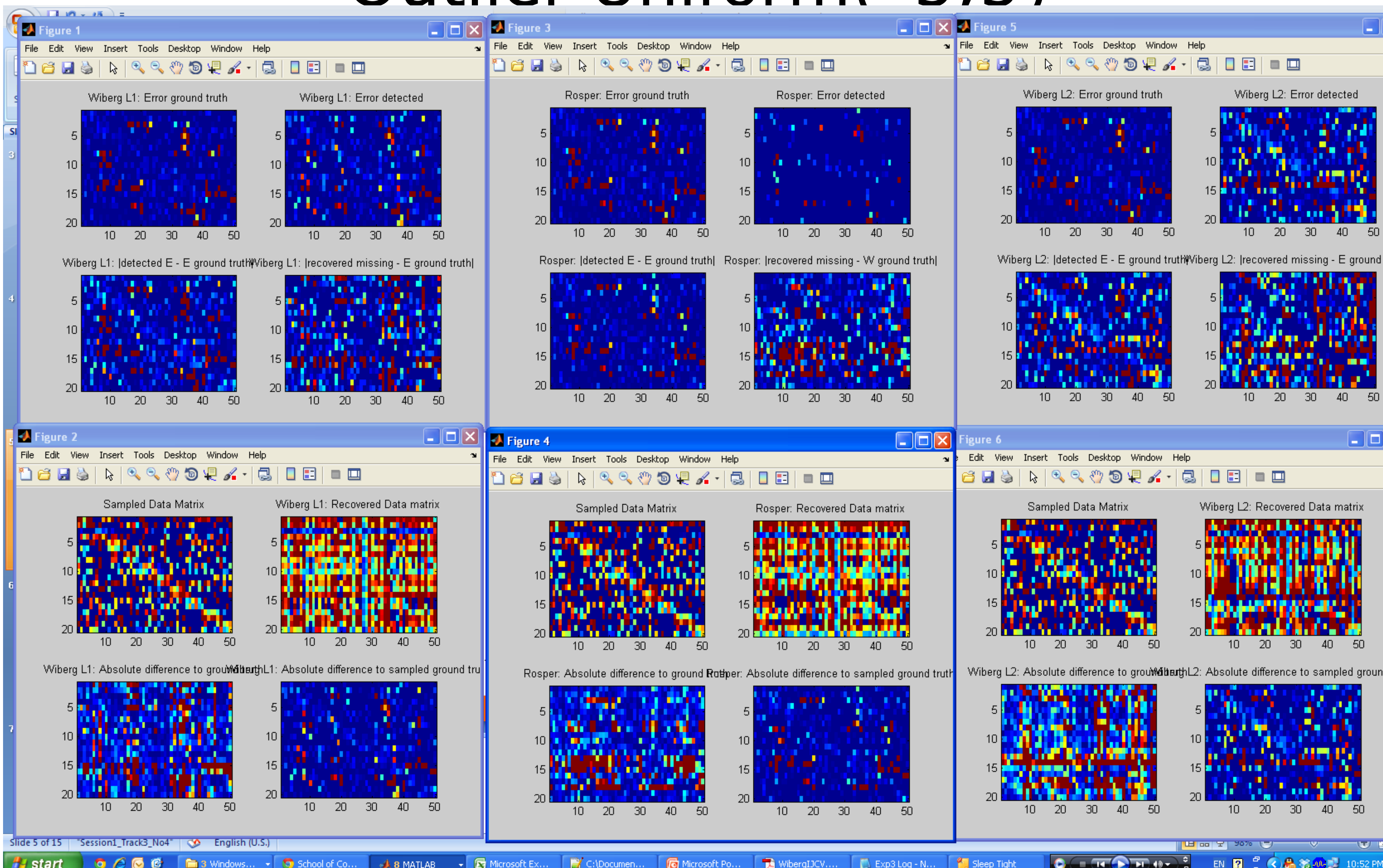
Dense Gaussian Noise



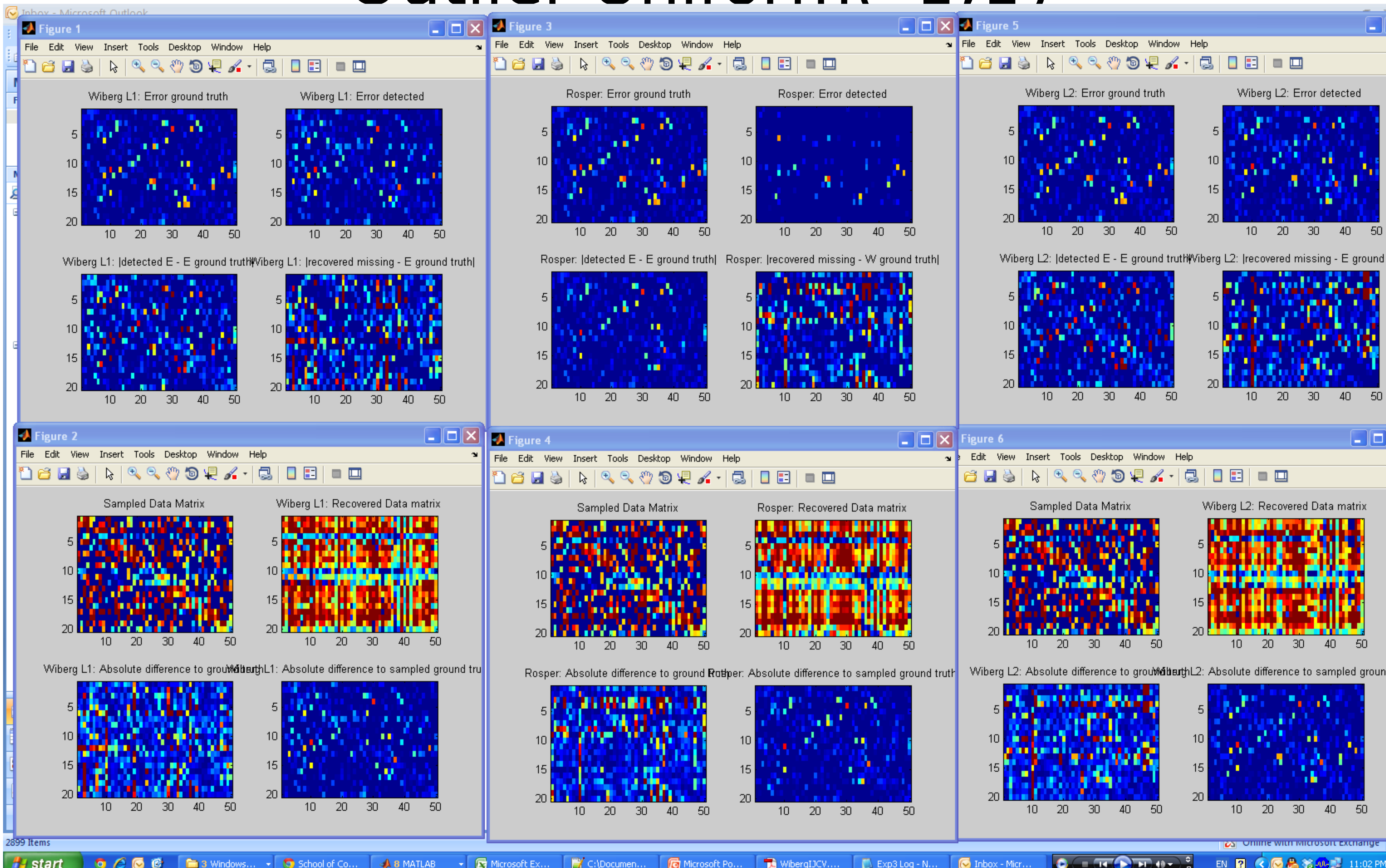
Dense Gaussian Noise



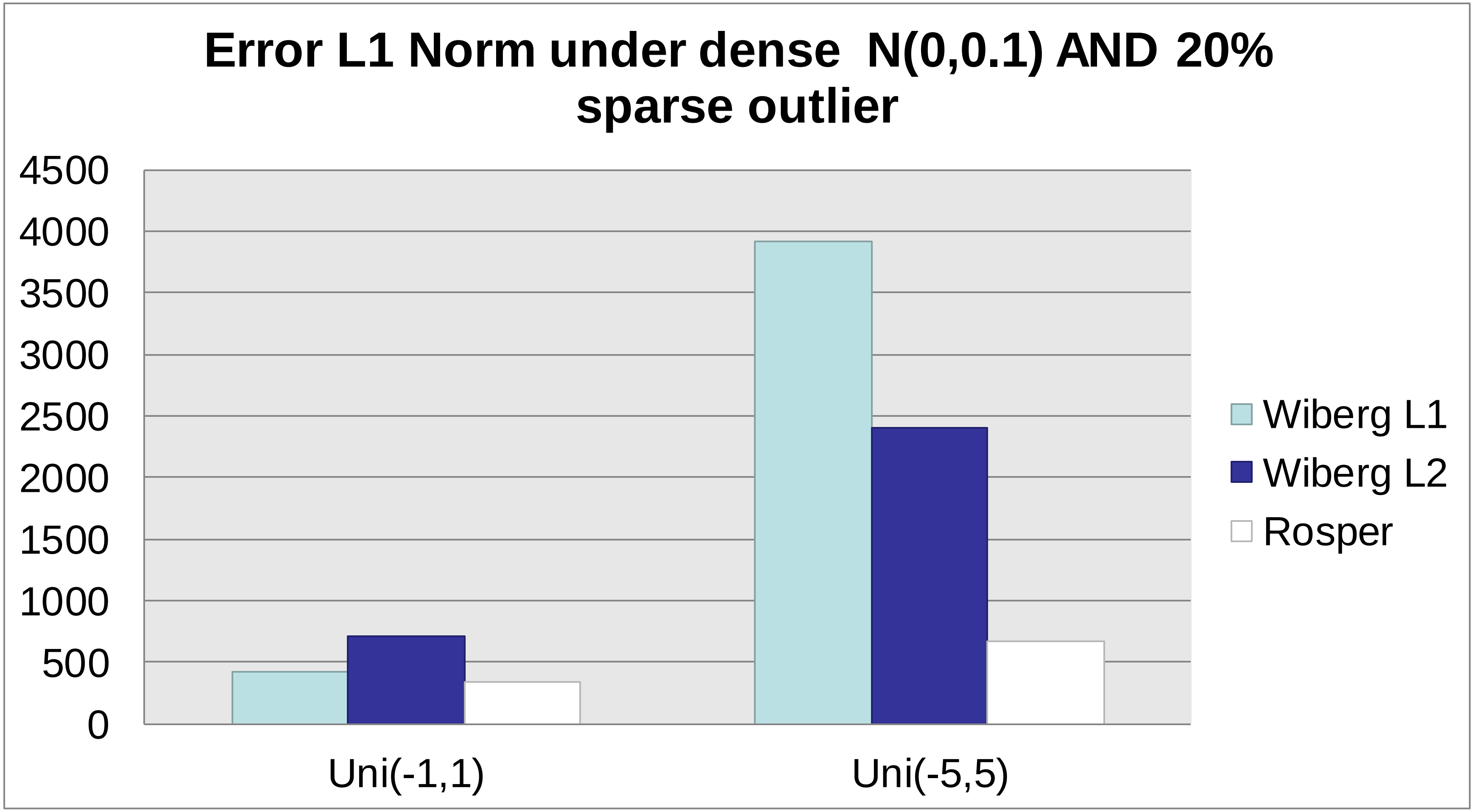
Experiment 11:Dense N(0,0.1)+0.2* Outlier Uniform(-5,5)



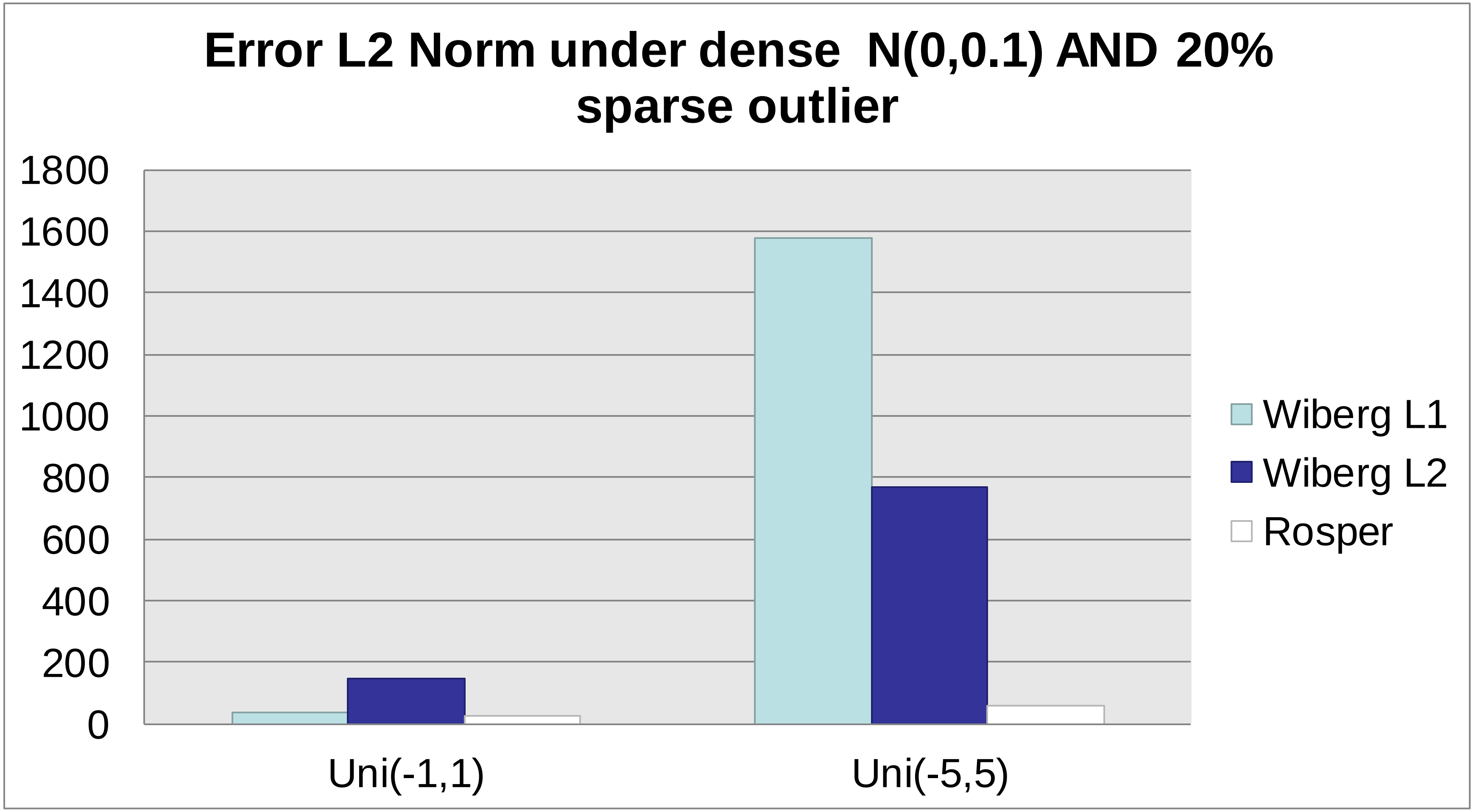
Experiment 12:Dense N(0,0.1)+0.2* Outlier Uniform(-1,1)



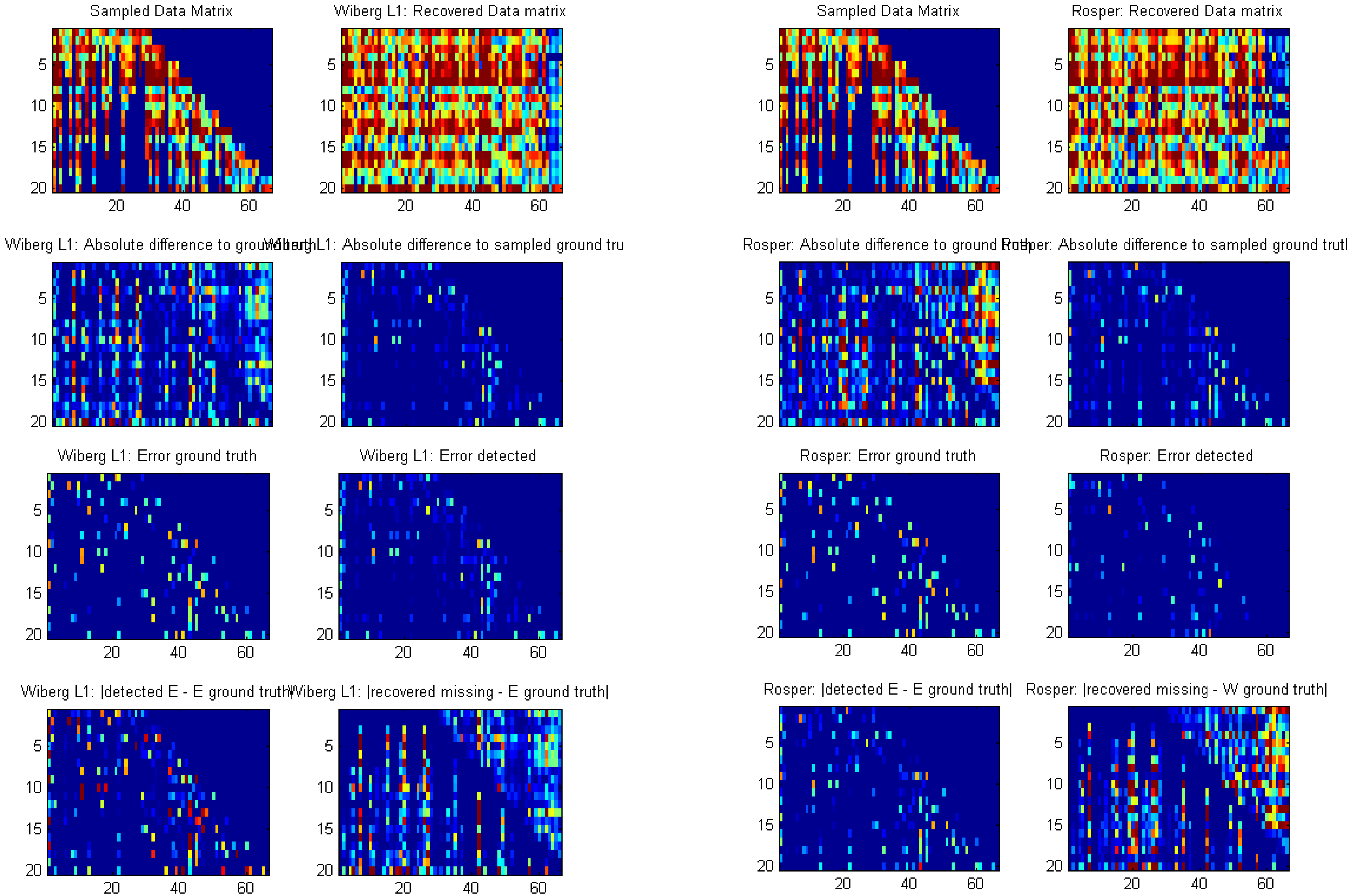
Dense Gaussian Noise + Sparse Outlier



Dense Gaussian Noise + Sparse Outlier



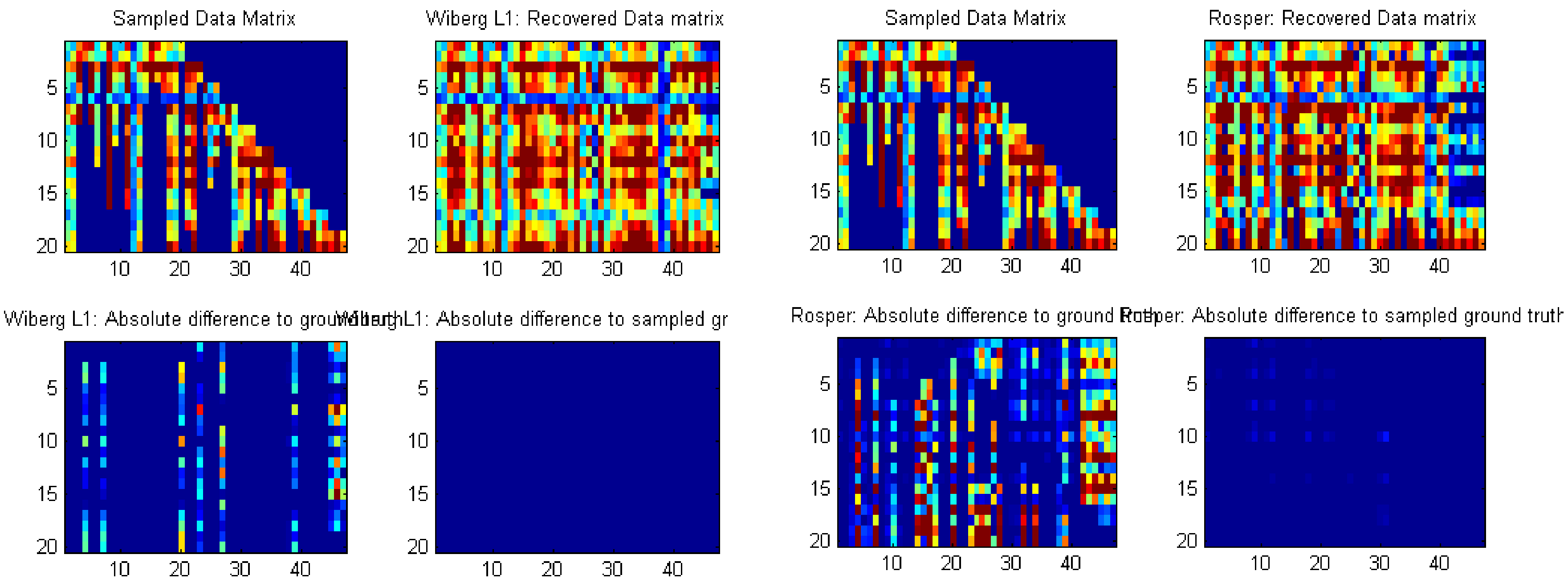
Diagonal band shaped data matrix



Diagonal band shaped data matrix

EXP1: 20% Uniform Error **0.45 Sample Rate** **[m n r]=[20 66 4]**	**Wiberg L1**	**Rosper**
# of iterations	17	2027
Running Time	6 min 36 sec	9 sec
|Recovered E - E_ground_truth|	106.72	52.55
|Recovered Missing - W_ground_truth|	253.35	397.26
L1 Norm: |Wgnd - Wr|	304.35	464.8
L2 Norm: |Wgnd - Wr|	16.38	25.53
L1 Norm: |mask(Wgnd - Wr)|	51	67.54
L2 Norm: |mask(Wgnd - Wr)|	4.55	5.2

Diagonal band shaped data matrix



Banded Results table

EXP2: 0 Uniform Error **0.43 Sample Rate** **[m n r]=[20 56 4]**	**Wiberg L1**	**Rosper**
# of iterations	35	5675
Running Time	6 min 59 sec	23 sec
|Recovered E - E_ground_truth|	<0.01	0.015
|Recovered Missing - W_ground_truth|	61.6	315.6
L1 Norm: |Wgnd - Wr|	61.61	322.02
L2 Norm: |Wgnd - Wr|	6.22	20.87
L1 Norm: |mask(Wgnd - Wr)|	<0.01	6.46
L2 Norm: |mask(Wgnd - Wr)|	<0.01	0.56

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