

RESULTS, CONCLUSIONS, AND FUTURE WORK*

Anthony Austin
Jose Garcia
Stephen Jong
Gilberto Hernandez

This work is produced by OpenStax-CNX and licensed under the
Creative Commons Attribution License 3.0[†]

Abstract

This module is part of a collection of modules for a class project on matrix completion techniques for the sensor network localization problem done for the Fall, 2009 offering of Prof. Baraniuk's ELEC 301 course at Rice University.

1 Results, Conclusions, and Future Work

1.1 Random Knock-Out Trials

The results from the simulations for the random knock-out runs are displayed in the figures below, which depict the average relative Frobenius-norm error over 25 trials versus fraction of unknown entries.

*Version 1.1: Dec 16, 2009 3:46 pm +0000

[†]<http://creativecommons.org/licenses/by/3.0/>

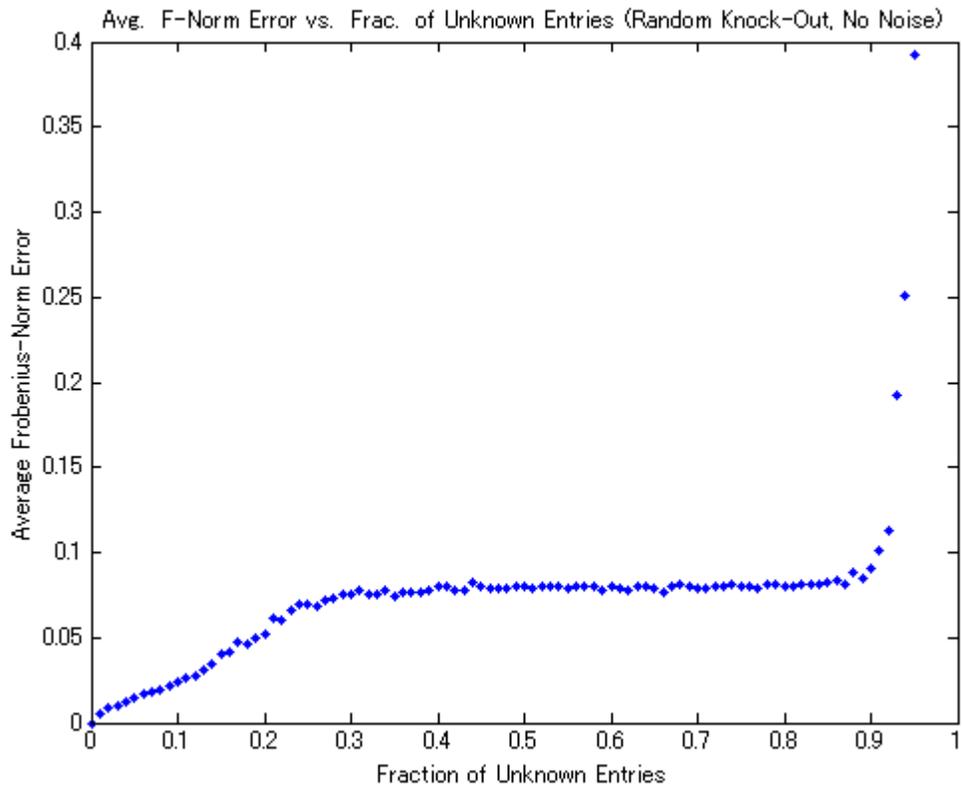


Figure 1: Simulation results for random knock-out trials with no noise.

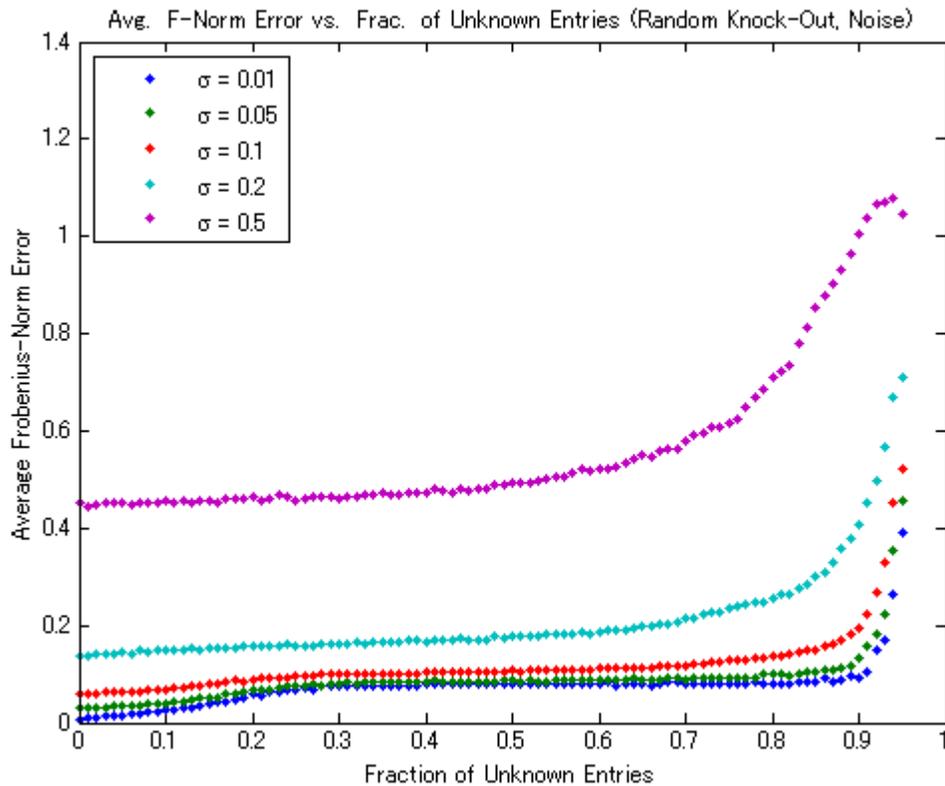


Figure 2: Simulation results for random knock-out trials with noise present.

As these two figures illustrate, the results for the random knock-out trials were quite good. As expected, as the fraction of unknown entries becomes large, the error eventually becomes severe, while for very low fractions of unknown entries, the error is extremely small. What is amazing is that for moderate fractions of unknown entries the algorithm still performs remarkably well, and its performance doesn't degrade much by the loss of a few more entries: the graphs are nearly flat over the range from 0.3 to 0.8! As the second figure shows (and as might be imagined), noise only makes the error worse; however, the plot also shows that the algorithm is reasonably robust to noise in that perturbations of the distance data by small amounts of noise don't become magnified into massive errors.

As an example, consider Figure 3 below, which displays the results of a typical no-noise random knock-out run with knock-out probability 0.5. On the left is a plot of the sparsity pattern for the incomplete matrix. A blue dot represents a known entry, while a blank space represents an unknown one. On the right is a plot of what the network looks like after being reconstructed using multidimensional scaling. Observe that the red circles for the network corresponding to the network generated by the completed matrix enclose the blue dots of the original network's structure quite well, indicating that the match is very good.

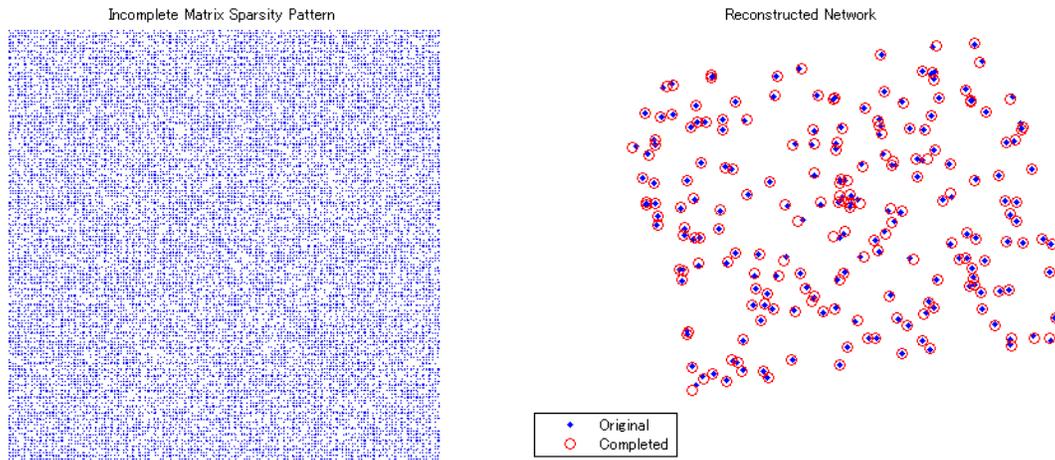


Figure 3: Results from a typical no-noise random knock-out trial with a knock-out probability of 0.5. Left: Sparsity pattern for the incomplete matrix. Right: Overlay figure demonstrating degree of agreement between the original network and the network generated from the completed matrix.

For an illustration of how the results look with noise, see Figure 4 below. This figure shows the results of a typical noise-present random knock-out run with knock-out probability of 0.5 and noise standard deviation 0.05. The agreement in the reconstructed network is not as good as it was for the no-noise case, but the points of the reconstructed network are “clustered” in the right locations, and some of the prominent features of the original network are present in the reconstructed one as well.

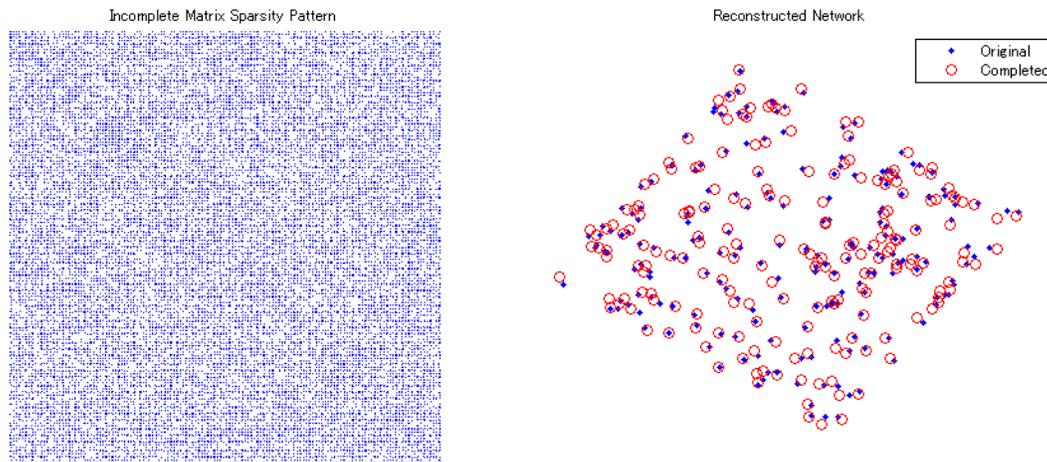


Figure 4: Results from a typical noise-present random knock-out trial with a knock-out probability of 0.5 and a noise standard deviation of 0.05. Left: Sparsity pattern for the incomplete matrix. Right: Overlay figure demonstrating degree of agreement between the original network and the network generated from the completed matrix.

1.2 Realistic Knock-Out Trials

The figures below, which show the results for the realistic knock-out trials, are similar to those above except that they plot the average relative Frobenius-norm error over 25 trials versus maximum radius as opposed to fraction of unknown entries.

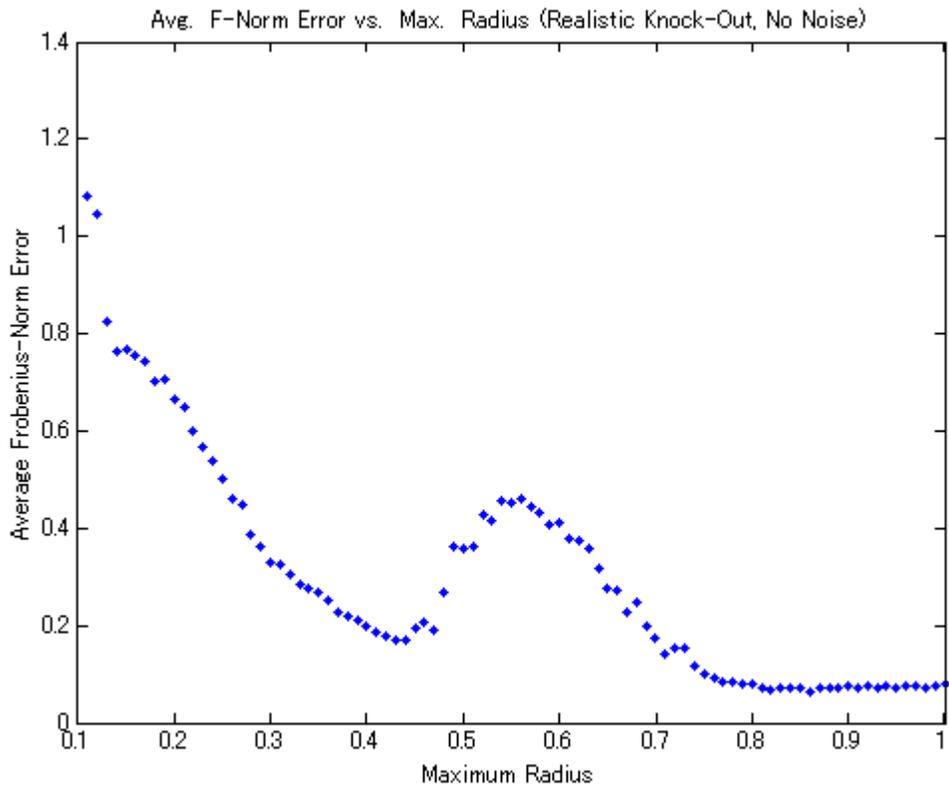


Figure 5: Simulation results for realistic knock-out trials without noise.

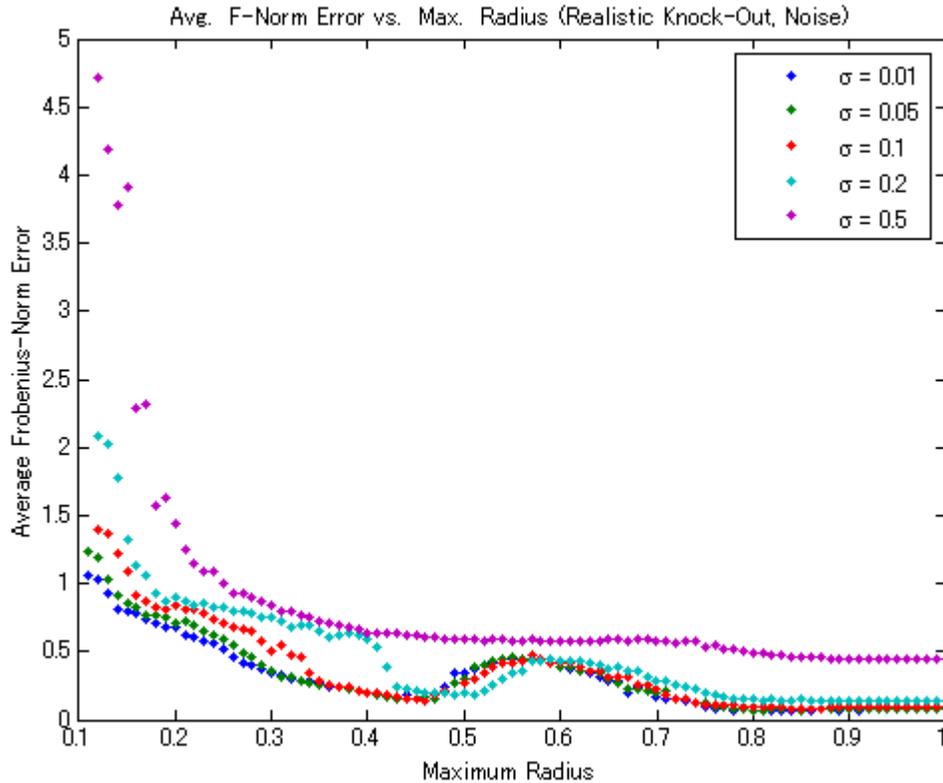


Figure 6: Simulation results for realistic knock-out trials with noise present.

The most salient feature of these graphs is the odd “hump” that appears from radius values of about 0.5 to 0.7, even in the no-noise case. Over this range, despite the fact that the radius is growing (meaning that more pairwise distances are known), the error in the completed matrix is actually becoming worse rather than better, which seems to contradict the excellent results discussed above for the random knock-out case. At the time of this writing, we are still unsure as to why this “hump” appears; however, we suspect that it may have something to do with the OptSpace algorithm itself because when we run the same experiment using the SVT algorithm of Candès, the hump does not appear, as the figure below shows. (Note that, nevertheless, OptSpace tends to produce less error than SVT, even over the offending range of radii.)

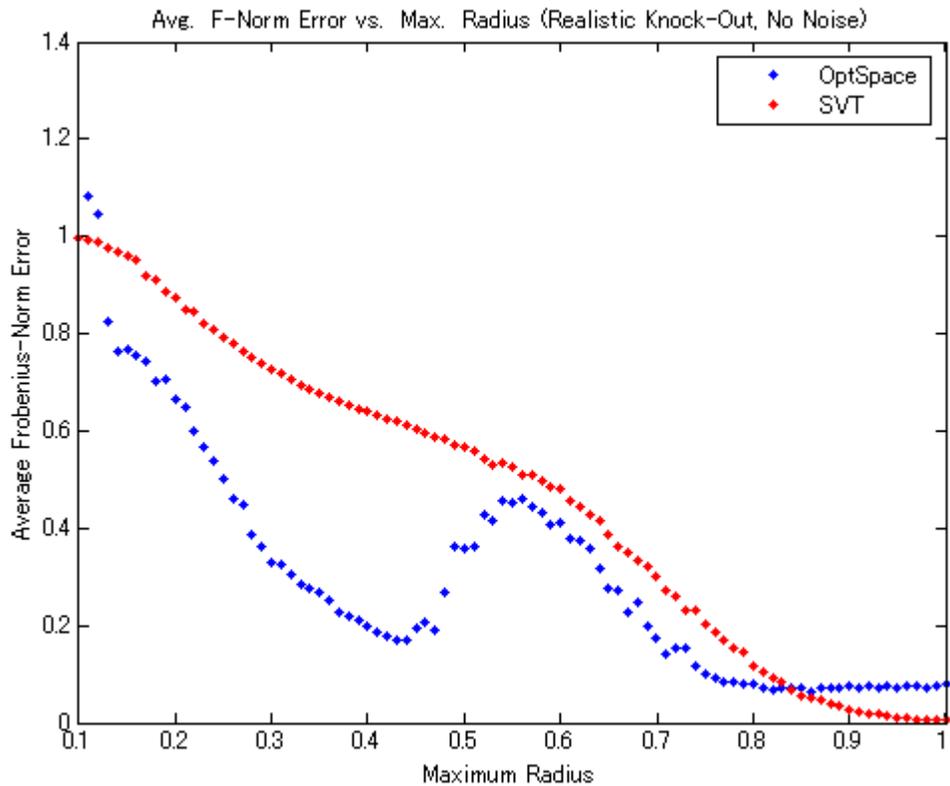


Figure 7: OptSpace performance vs. SVT performance for realistic entry knock-out without noise. SVT does not display a “hump,” but OptSpace generally returns better error values.

Perhaps more important than the “hump,” however, is the fact that the scales on the axes of the above graphs alone are enough to demonstrate that the performance of the method in the realistic knock-out case is decidedly worse than that for the random knock-out case. For example, consider the figure below, which shows the results of a typical no-noise, realistic knock-out trial with a maximum radius of 1. For this particular trial, over 97 percent of the pairs are known. The reconstructed network matches the original quite well near the “center” of the network, but at the edges, the match becomes much worse. This behavior is not exhibited at all by random knock-out trials for comparable fractions of unknown entries, as the picture at the bottom of the figure illustrates, which was generated from a non-noise random knock-out trial in which 90 percent of the pairs were known.

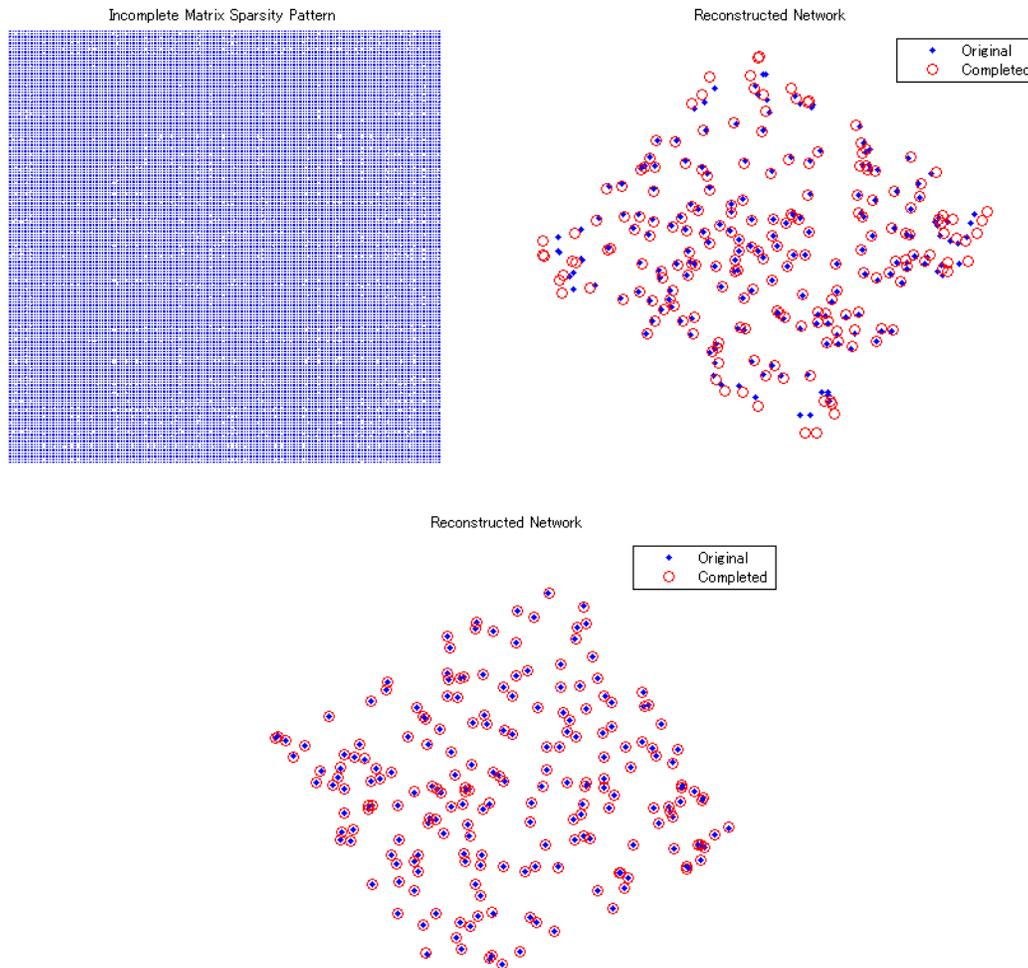


Figure 8: Results from a typical non-noise realistic knock-out trial with a maximum radius of 1. Top-Left: Sparsity pattern for the incomplete matrix. Top-Right: Overlay figure demonstrating degree of agreement between the original network and the network generated from the completed matrix. Bottom: Typical results from a non-noise random knock-out trial with knock out probability of 0.1 (90% of distance pairs are known).

Shrinking the radius only makes matters worse, as the next figure illustrates. The maximum radius here is $\sqrt{2}/2$. Around 77 percent of the distance pairs are known, and yet the match is terrible.

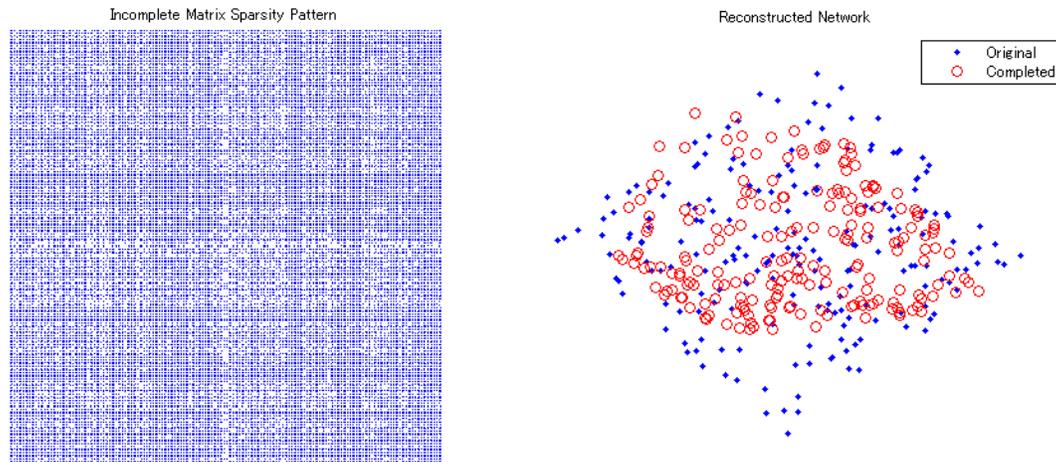


Figure 9: Results from a typical non-noise realistic knock-out trial with a maximum radius of $\sqrt{2}/2$. Left: Sparsity pattern for the incomplete matrix. Right: Overlay figure demonstrating degree of agreement between the original network and the network generated from the completed matrix.

Adding noise only makes the results even worse. At first glance, this behavior appears to be inexplicable; however examining the sparsity patterns of the incomplete matrices reveals an interesting fact: the entry knock-out in the realistic case is far from being “random!” The sparsity patterns for the realistic knock-out matrices reveal clear patterns of lines in their knocked-out entries that are not present in those for random knock-out cases. This unintended regularity of entry selection violates the assumption made in all of the matrix completion literature that the known entries are taken from a uniform sampling of the matrix, so it would seem that none of the theoretical results that have been derived apply in this case.

1.3 Conclusions

Our results show that matrix completion presents a viable means of approaching the sensor network localization problem under the assumption that the known pairs of distances come from a uniform sampling of the distance matrix. Under these conditions, matrix completion provides excellent network reconstruction and is fairly robust to noise. Unfortunately, its performance in the more realistic case in which distance information will be excluded or included based on a maximum possible distance over which two sensors can communicate leaves much to be desired.

1.4 Future Work

With more time on this project, we would like to have explored the following questions further:

- What is the true origin of the mysterious “hump?” If it really is due to OptSpace as the above seems to suggest, is there a way to modify the OptSpace algorithm to get it to go away?
- What is the fundamental reason that the realistic knock-out trials did not work? Is there a way to get them to work better? (Perhaps something like permuting the distance entries in the matrix around

to make the sampling pattern apparently more random would do the trick. If the permutations are stored somewhere, they can be undone after the matrix is completed if necessary.)

- The experiment worked pretty well in two dimensions, at least for the random knock-out case. Will three dimensions show results that are any different?

References