Compressively sensed complex networks

J. Ray, A. Pinar and D. Dunlavy
Sandia National Laboratories

Acknowledgements: ASCR, Office of Science, Dept. of Energy.
Introduction

• **Aim:** To develop low dimension parametric (deterministic) models of complex networks
  – Use compressive sensing (CS) and multiscale analysis to do so
    • Exploit the structure of complex networks (some are self-similar under coarsening)
  
• **Motivation:**
  – Graphs often form the understructure over which dynamics occur
    • e.g., chemical reactions, epidemiological processes, cascading failures, etc
  – Dynamics are easy to observe, but the graphical structure unknown
  – A low-dimension model of a graph allows its “discovery” by fitting to data
    • Inverse problem, network discovery, estimation of graphs etc
Today’s talk

• Can compressive sensing actually work?
  – Under what circumstances?

• Some assumptions/characteristics of the networks
  – Networks will be small – O(100) nodes
    • In inverse problems, not enough info to fit detailed models
  – Networks will be assumed to have densely connected “cliques” of different sizes
    • Leads to “blocky” adjacency matrices

• Outline
  – Compressive sensing – what is it?
    • Sampling technique
    • Reconstruction technique
  – Test cases
    • 2 synthetic networks
    • Results – different levels of sampling/order reduction and reconstruction fidelity
What is compressive sensing?

• A technique to efficiently encode/decode a random vector $x$ of length $N$
  – $x$ can be a signal, a time-series
  – The process is lossy – the decoding is approximate
• Efficiency of representation rests on the presumption that the information content of the signal is small
  – i.e., a sparse representation exists for $x$
  – In conjunction with efficient sampling, only a few samples are needed
• “Sampling” a signal means projection on a sampling basis set $\psi_i$
  \[ y = \Psi x \]
  – $y$ is the “signature”; $y_i$ are the projections of $x$
  – Under certain conditions $\text{size}(y) \sim \log(N)$
When and why is $Y$ compressive?

- If $x$ can be described sparsely in an orthogonal basis set (e.g., wavelets), $\Phi$, then the basis weights can be sampled directly

$$x = \Phi s, \quad y = \Psi s$$

- Where only $K$ elements of $s$ are non-zero. $K \ll \text{size}(s) = N$

- An efficient sampling of $x$ collects information on all elements of $s$ per projection

  - Can be done if $\psi_i$ are random vectors
    - e.g., chosen from a high dimensional sphere (uniform spherical ensemble)

  - $\Psi$ is then an orthogonal matrix

- Under these conditions

$$M = \text{size}(y) \geq cK \log(N / K) \ll N$$
Decoding - reconstructing $x$ from $Y$

• Canonically, decoding is performed via $l_1$ minimization

$$\min \|s\|_1 \text{ subject to } \|y - \Psi s\|_2 < \varepsilon$$

— Exact solutions (e.g., basis pursuit) too computationally expensive, so usually an approximate form is solved in practice

• Our algorithm, called StOMP (Stagewise Orthogonal Matching Pursuit)
  — Donoho et al, 2006 (preprint)
  — Iteratively finds the non-zero elements of $s$
  — Number of iterations are bounded
    • But assumes that $s$ is sparse
  — Computational cost comes from the pseudoinverse of $\Psi$
  — Suitable for large problems
Extending Compressive Sensing to networks

- Based on the CS of adjacency matrices
- We expect that the rows/columns of an adjacency matrix can be re-ordered to create a “blocky” adjacency matrix
  - Alternatively, node in a clique should have similar node-ids
- Networks showing self-similarity will show structure within the blocks
- Exploiting multi-resolution:
  - Decompose the adjacency matrix on a wavelet basis
    - In our case, Haar wavelets
  - Haar wavelet coefficients stored and treated hierarchically
    - Resolution by resolution, with no inter-resolution dependence modeled or exploited
  - Non-zero wavelet coefficients at each resolution will be sparse
  - Sampling and reconstruction too are performed hierarchically
A graph and its multiscale decomposition - I

The network

Adjacency matrix

256 nodes, $256^2$ wavelet coefficients
A graph and its multiscale decomposition - II

Wavelet coefficients from the Haar transform of the adjacency matrix

Histogram of coefficient values
(log10(Frequency))

Histogram (log10 scale) of the magnitude of wavelet coefficients

Wavelet coefficients at different resolutions

256 nodes, $256^2$ wavelet coefficients
Sampling issues

• The number of wavelet coefficients at level \( l \) is \( 4^l \)

• How many samples per level?
  – Generally expressed as a fraction of \( 4^l \)
  – Should this fraction vary with levels

• Some observations
  – Sampling has to be minimal at high resolutions
    • i.e. fine detail cannot be captured
    • Emphasis on “blockiness”, structure should be evident at a certain resolution
  – All wavelet coefficients at coarser resolutions can be retained

• So CS can be expected to work for graphs that are somewhat dense
  – Will not work for very sparse graphs
Tests

- 2 synthetic directed graphs, of 256 nodes. Average degree of 23 & 17
- Sample (to various degrees) and reconstruct
- Performance wrt number of samples and sparsity of graph
Test 1(a) – fine sampling (60%)

- Network I, with 256 nodes and 5914 edges
  - Needs $2N_{\text{edges}}$ to store
- Reconstruction with ~60% sampling

<table>
<thead>
<tr>
<th></th>
<th>Original</th>
<th>Reconstruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of samples</td>
<td>11,824</td>
<td>7463 (~60%)</td>
</tr>
<tr>
<td>Average degree</td>
<td>23.1</td>
<td>27.2</td>
</tr>
<tr>
<td>Fiedler Value</td>
<td>1.85</td>
<td>1.91</td>
</tr>
<tr>
<td>No. edges</td>
<td>5914</td>
<td>6924</td>
</tr>
</tbody>
</table>

Degree distribution

Eigenvalues of the Laplacian
Test I(a) – comparison of nets

Original

Reconstruction
Test I(a) – analysis of differences

- Difference of adjacency matrices
  - Small differences around the blocks
  - Normalized error (Frobenius norm) ~ 42%
  - Red edge: false negative; Blue edge: false positive
Test I(b) – coarse sampling (25%)

- Network I, with 256 nodes and 5914 edges
  - Needs $2N_{\text{edges}}$ to store
- Reconstruction with ~ 25% sampling

<table>
<thead>
<tr>
<th></th>
<th>Original</th>
<th>Reconstruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of samples</td>
<td>11,824</td>
<td>2478 (~25%)</td>
</tr>
<tr>
<td>Average degree</td>
<td>23.1</td>
<td>44.6</td>
</tr>
<tr>
<td>Fiedler Value</td>
<td>1.85</td>
<td>1.81</td>
</tr>
<tr>
<td>No. edges</td>
<td>5914</td>
<td>11424</td>
</tr>
</tbody>
</table>

Degree distribution

Eigenvalues of the Laplacian
Test I(b) – analysis of differences

- Difference of adjacency matrices
  - Significant structural differences
  - Normalized error (Frobenius norm) ~ 100%
  - Red edge: false negative; Blue edge: false positive
Test II - a sparser net

- Network II, with 256 nodes and 4426 edges
- Reconstruction with ~ 85% sampling

<table>
<thead>
<tr>
<th></th>
<th>Original</th>
<th>Reconstruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of samples</td>
<td>8852</td>
<td>7589 (~85%)</td>
</tr>
<tr>
<td>Average degree</td>
<td>17.29</td>
<td>20.1</td>
</tr>
<tr>
<td>Fiedler Value</td>
<td>1.08</td>
<td>1.54</td>
</tr>
<tr>
<td>No. edges</td>
<td>4426</td>
<td>5144</td>
</tr>
</tbody>
</table>

Degree distribution

Eigenvalues of the Laplacian
Test II – comparison of nets

Original

Reconstruction
Test II – analysis of differences

- Difference of adjacency matrices
  - Small differences around the blocks
  - Normalized error (Frobenius norm) ~ 40%
  - Red edge: false negative; Blue edge: false positive
Summary of the tests

• Define: Average link probability = average degree / # of nodes

• For an average link probability of around 10%:
  – 60% sampling gives excellent reconstruction
  – 25% sampling leads to over estimation of average degree
    • i.e., the reconstructed graph is very coarse & lacks detail

• For an average link probability of around 7%:
  – The technique requires too many samples (~85%) and is not competitive

• In general, matching the eigenvalue spectrum is easy
  – Fiedler value less so, but getting to +/- 10% is possible

• Matching the degree distribution is harder
  – 25% sampling does not do it
  – 60% or higher does it, depending upon the average link probability
Summary and Conclusions

• CS provides a new way of sampling and reconstructing networks
• Approach based on multiresolution decomposition of the adjacency matrix and its efficient sampling
• Requires preprocessing of the adjacency matrix to make it “blocky”
  – Biggest (combinatorial) algorithm challenge.
• Current CS reconstruction algorithm makes no use of the structure of a graph – very general (and so not very efficient/customized)
  – Other model-based CS techniques exist, but not yet adapted to networks
  – Obvious starting point for future work to increase the efficiency of reconstruction