Octave routines for network analysis

GB

July 9, 2014

Contents

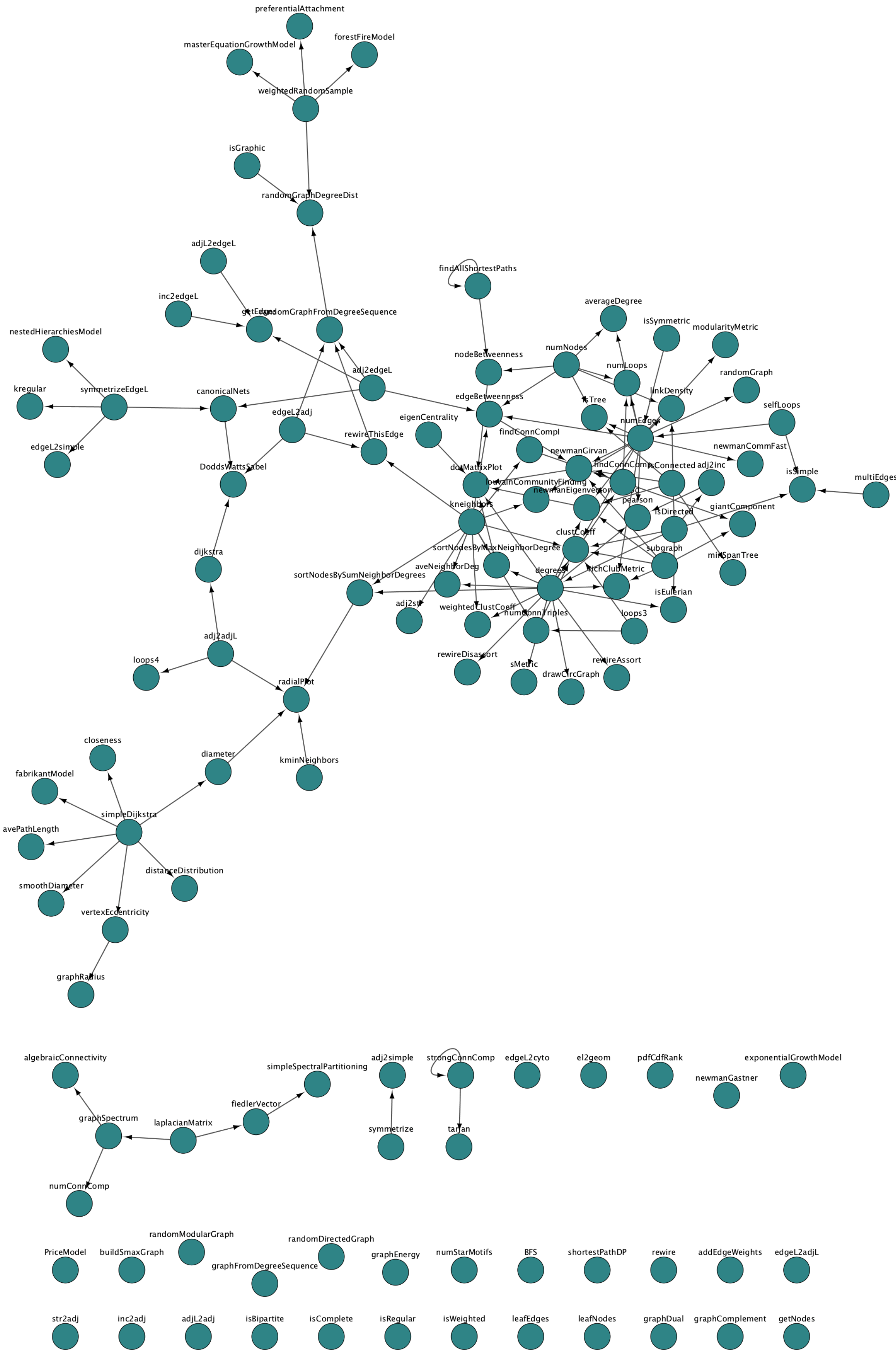

Figure 1: Graph of functional interdependencies in this toolbox. An edge points from routine A to routine B if routine A is called within routine B. For example, *isConnected.m* is called within *minSpanTree.m*

0 About this toolbox

(This is a copy of the README file.)

octave-networks-toolbox: A set of graph/networks analysis functions in Octave, 2012-2014

Quick description
This is a repository of functions relevant to network/graph analysis, organized by functionality. These routines are useful for someone who wants to start hands-on work with networks fairly quickly, explore simple graph statistics, distributions, simple visualization and compute common network theory metrics.

History
The original (2006-2011) version of these routines was written in Matlab, and is still hosted by strategic.mit.edu (http://strategic.mit.edu/downloads.php?page=matlab_networks). The octave-networks-toolbox inherits the original BSD open source license and copyright, provided at the end of this file. Many of the routines might still be compatible with Matlab. For Octave/Matlab differences, see http://en.wikibooks.org/wiki/MATLAB_Programming/Differences_between_Octave_and_MATLAB.

Installation
The code currently runs on GNU Octave Version 3.4.0 with Gnuplot 4.2.5. No specific library installation necessary. Interdependencies between functions are well-documented in the function headers. The routines can be called directly from the Octave prompt, either in the same directory or from anywhere if the toolbox folder is added to the path. For example:

```
# running numNodes.m
octave-3.4.0:1> numNodes([0 1 1; 1 0 1; 1 1 0])
ans =  3
```

Authorship
This code was originally written and posted by Gergana Bounova. It is undergoing continuous expansion and development. Thank you for the many comments and bug reports so far! Contributions via email are usually given tribute to in the function header. Collaborators are very welcome. Contact via github/email for comments, questions, suggestions, corrections or simply fork.

Organization
The functions are organized in 11 categories: basic routines, diagnostic routines, conversion routines, centralities, distances, simple motif routines, linear algebra functions, modularity routines, graph construction models, visualization and auxiliary. These categories reflect roles/functionality and topics in the literature, but they are arbitrary, and mostly used for documentation purposes.

Documentation
Documentation is available in this Functions Manual. The manual contains general background information, function headers, code examples, and references. For some functions, additional background, definitions or derivations are included.

Citation
If you want to refer to this code as a citation, you can use DOI: 10.5281/zenodo.10778 (https://zenodo.org/record/10778).

1 Basic network routines

1.1 Basic network theory

A **graph** is a set of nodes, and an associated set of links between them.

Networks are instantiations of graphs. They often represent real world systems that can be modeled as a set of connected entities.

Network theory is a modern branch of **graph theory**, concerned with statistics on practical instances of mathematical graphs. Graph theory and network theory references are abundant. Social science is probably the most recent instigator of the trend to see the world as a network. In 1967, Milgram conducted his famous small world experiment [1], and found that Omahans are on average six steps away by acquaintance from Bostonians. Other prominent first sources are Price's work on the graph of scientific citations in 1965 [2] and in 1998, Watts and Strogatz's paper on dynamics of small-world networks [3].

Nowadays, there is no shortage of books and reviews on networks. Below is a non-exhaustive list of good reads [4] [5] [6] [7].

- S. Wasserman and K. Faust, Social network analysis, Cambridge University Press, 1994
- Duncan J. Watts, Six degrees: The science of a Connected Age, W. W. Norton, 2004
- M. E. J. Newman, The structure and function of complex networks, SIAM Review 45, 167-256 (2003)
- Alderson D., Catching the Network Science Bug: ..., Operations Research, Vol. 56, No. 5, Sep-Oct 2008, pp. 1047-1065

Here are some basic notions about graphs that are useful to understand the routines in Section **??**.

Figure 1 illustrates a general **directed** graph. The nodes are functions from this toolbox. An edge points from function A to function B if *function A is called within function B*. For example, *strongConnComp* is used within *tarjan*. Notice, also that *strongConnComp* points to itself, i.e. *strongConnComp* contains a recursion. Stand-alone functions, that use no other function, are **single nodes** in the graph, such as *leafNodes*, *getEdges* and *graphDual*.

A **directed graph** is a graph in which the links have a direction. In the functions graph one function can call another, but the call is usually not reciprocated.

A **single node** is a node without any connections to other nodes. *graphDual* is an example of a single node in Figure 1.

A **self-loop** is an edge which starts and ends at the same node. (*strongConnComp*→*strongConnComp*) is an example of a self-loop.

Multiedges are two or more edges which have the same origin and destination pair of nodes. This can be useful in some graph representations. In the functions graph this is equivalent to some function being called twice inside another function.

Basic graph statistics are the **number of nodes** (n) and the **number of edges** (m). The functions graph has 118 nodes and 125 edges.

The **link density** is derived directly from the number of nodes and number of edges: it is the number of edges, divided by the maximum possible number of edges.

$$density = \frac{m}{n(n-1)/2} \tag{1}$$

For the functions graph, the link density is about 0.0181. Note that equation 1 is valid for undirected graphs only.

The **average nodal degree** is the average number of links per node. This is calculated as $2m/n$ (every edge is counted twice towards the total sum of degrees).

$$average\ degree = \frac{2m}{n} \tag{2}$$

The functions graph has 2.12 links per function on average.

A graph S is a **subgraph** of graph G, if the set of nodes (and edges) of S is subset of the set of nodes (and edges) of graph G.

A **disconnected** graph is a graph in which there are two nodes between which there exists no path of edges. In the functions graph there is no path between *rewire* and *subgraph*. So the functions graph is disconnected. Disconnected graphs consist of multiple connected components. The largest connected component (in number of nodes) is usually called the **giant component**. The giant component in Figure 1 has 80 functions. There are also one connected components of 6 functions, two 2-node components and 28 isolated nodes (functions that do not call or are not called within other functions).

In the context of **directed graphs**, the notion of strong and weak connectivity is important. A **strongly connected graph** is a graph in which there is a path from every node to every other node, where paths respect link directionality. In Figure 1, for example, there is a path from *strongConnComp* to *tarjan*, but no path in reverse. Therefore, the component (*strongConnComp*,*tarjan*) is not strongly connected. If, however, link directionality is disregarded, this subgraph is certainly connected. A **weakly connected graph** or subgraph is a graph which is connected if considered as undirected, but not connected if link directionality is taken into account. So the two-node subgraph (*strongConnComp*,*tarjan*) is definitely weakly connected.

References

[1] Milgram's small world experiment; source: http://en.wikipedia.org/wiki/Small_world_experiment, last accessed: Sep 23, 2012

[2] D.J. de S. Price, Networks of scientific papers, Science, 149, 1965

[3] D. Watts and S. Strogatz, Collective dynamics of 'small-world' networks, Nature 393, 1998

[4] S. Wasserman and K. Faust, Social network analysis, Cambridge University Press, 1994

[5] Duncan J. Watts, Six degrees: The science of a Connected Age, W. W. Norton, 2004

[6] M. E. J. Newman, The structure and function of complex networks, SIAM Review 45, 167-256 (2003)

[7] Alderson D., Catching the Network Science Bug: ..., Operations Research, Vol. 56, No. 5, Sep-Oct 2008, pp. 1047-1065

[8] Tarjan, R. E. , Depth-first search and linear graph algorithms, SIAM Journal on Computing 1 (2): 146-160, 1972

[9] Wikipedia description of Tarjan's algorithm; source: http://en.wikipedia.org/wiki/Tarjan's_strongly_connected_components_algorithm, last accessed: Sep 23, 2012

[10] Erdös, P. and Gallai, T. Graphs with Prescribed Degrees of Vertices [Hungarian]. Mat. Lapok. 11, 264-274, 1960.

[11] Alderson, Li, Diversity of graphs with highly variable connectivity, Phys. Rev. E 75, 046102 (2007)

[12] Vittoria Colizza, Alessandro Flammini, M. Angeles Serrano, Alessandro Vespignani, Detecting rich-club ordering in complex networks, Nature Physics 2, 110-115 (2006)

[13] Wikipedia entry on eigenvector centrality; source: http://en.wikipedia.org/wiki/Centrality#Using_the_adjacency_matrix_to_find_eigenvector_centrality, last accessed: October 2, 2012

[14] Wikipedia article on node betweenness; source: http://en.wikipedia.org/wiki/Betweenness_centrality, last accessed: September 28, 2012

[15] M. E. J. Newman, M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69, 026113 (2004)

[16] A. Barrat, M. Barthélemy, R. Pastor-Satorras, and A. Vespignani, The architecture of complex weighted networks, PNAS March 16, 2004 vol. 101 no. 11, 3747-3752

[17] M. E. J. Newman, Assortative mixing in networks, Phys. Rev. Lett. 89, 208701 (2002)

[18] Lun Li, David Alderson, Reiko Tanaka, John C. Doyle, Walter Willinger, Towards a Theory of Scale-Free Graphs: Definition, Properties, and Implications, Internet Math. Volume 2, Number 4 (2005), 431-523

[19] Guo, Chen, Zhou, Fingerprint for Network Topologies, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, ISSN1867-8211, Vol 5,Part 1, Springer Berlin Heidelberg 2009

[20] Bertsekas, Dynamic Programming and Optimal Control, Athena Scientific, 2005 (3rd edition)

[21] Leskovec, Kleinberg, Faloutsos, Graphs over Time: Densification Laws, Shrinking Diameters and Possible Explanations, KDD'05, 2005, Chicago, IL

[22] Mahadevan, Krioukov, Fall, Vahdat, Systematic Topology Analysis and Generation Using Degree Correlations, SIGCOMM '06 Proceedings

[23] I. Gutman, The energy of a graph, Ber. Math. Statist. Sekt. Forsch-ungszentram Graz. 103 (1978) 1-22.

[24] M. E. J. Newman, Finding community structure using the eigenvectors of matrices, Phys. Rev. E 74, 036104 (2006)

[25] M. E. J. Newman, Modularity and community structure in networks, PNAS June 6, 2006, vol. 103, no. 23, 8577-8582

[26] M. E. J. Newman, Fast algorithm for detecting community structure in networks, Phys. Rev. E 69, 066133 (2004)

[27] Blondel, Guillaume, Lambiotte, Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech. (2008) P10008

[28] M. E. J. Newman, Analysis of weighted networks, Phys. Rev. E 70, 056131 (2004)

[29] Erdös, Paul; A. Rényi, On the evolution of random graphs, Publications of the Mathematical Institute of the Hungarian Academy of Sciences 5: 17-61, 1960

[30] S. L. Hakimi, On Realizability of a Set of Integers as Degrees of the Vertices of a Linear Graph. I, Journal of the Society for Industrial and Applied Mathematics Vol. 10, No. 3 (Sep., 1962), pp. 496-506

[31] Molloy M., Reed, B. A Critical Point for Random Graphs with a Given Degree Sequence, Random Structures and Algorithms 6 , 161-179, 1995

[32] Clauset, Finding local community structure in networks, Phys. Rev. E 72, 026132 (2005)

[33] Dorogovtsev, Mendes, Evolution of Networks, Advances in Physics 2002, Vol. 51, No. 4, 1079-1198

[34] Gastner, Newman, Shape and efficiency in spatial distribution networks, J. Stat. Mech. (2006) P01015

[35] Fabrikant, Koutsoupias, Papadimitriou, Heuristically Optimized Trade-offs: A New Paradigm for Power Laws in the Internet, Automata, Languages and Programming, Vol. 2380, 2002

[36] Dodds, Watts, Sabel, Information exchange and the robustness of organizational networks, PNAS, vol. 100, no. 21, 12516-12521, October 14 2003

[37] Sales-Pardo, Guimerà, Moreira, Amaral, Extracting the hierarchical organization of complex systems, PNAS, vol. 104, no. 39, 15224-15229, September 25 2007

[38] Leskovec, Kleinberg, Faloutsos, Graph Evolution: Densification and Shrinking Diameters, ACM Transactions on Knowledge Discovery from Data (ACM TKDD), 1(1), 2007