CiGRAM Documentation

Release 1.0.0

James Gilbert

Mar 09, 2018

Contents:

CHAPTER 1

Generating networks

Generating networks in CiGRAM is straightforward.

```
from cigram import generate_graph
n = 1000
avg_k = 4.95
k = 1
graph, positions, communities = cigram_graph(n, avg_k, k)
```

Here n is the number of nodes, avg_k is the desired average degree and k is the number of communities. Note that the number of communities here is fixed at 1, so the graph will not generate artificial clusters.

The resulting returned tuple of `graph`, `positions` and `communities` is a networkx graph object, a dictionary of points upon a unit circle for each node, and a dictionary for the community membership of each vertex.

More complex parameters allow you to generate different heterogenous degree distributions. This is controlled by the parameters sigma_f and sigma_r which have an effect on the underlying probability space for the connections between nodes.

```
sigma_r = 0.8
sigma_f = 0.8
graph, positions, communities = cigram_graph(n, avg_k, k)
```

To generate networks with assortativity this can be specified with the parameter a (by default this is 0).

CHAPTER 2

LFR Benchmarks

CiGRAM also includes the generation of Lancichinetti–Fortunato–Radicchi (LFR) benchamrks. This was implemented due to issues found with the python implementation in NetworkX. This version is mostly a tidied up version of the original C++ code. To use:

```
from cigram import lfr_benchmark_graph

params = {
    'n': 10000,
    'average_degree': 10,
    'max_degree': 1000,
    'mu': 0.5,
    'tau': 2.0,
    'tau2': 2.0,
    'minc_size': 3,
    'maxc_size': 1000,
    'overlapping_nodes': 0,
    'overlapping_memberships': 1,
    'seed': 1337
}

graph, comms = lfr_benchmark_graph(**params)
```

CHAPTER 3

Introduction

CiGRAM is a Circular Gaussian Random Graph Generator written in python and C++. The objective of cigram is to generate large, complex networks with realistic properties such as high clustering, heterogenous degree distributions, community structure and assortativity.

These documents intend to describe how CiGRAM works, how you can generate networks with it and how you can use the network fitting package within this project to match the desired properties of real world networks. This documentation intends only to give a high level overview of the project. For more details please see the initial publication of CiGRAM:

Gilbert, J.P., 2015. A probabilistic model for the evaluation of module extraction algorithms in complex biological networks (Doctoral dissertation, University of Nottingham). http://eprints.nottingham.ac.uk/30524/

CHAPTER 4

Installation

Install with pip from pypi

```
pip install cigram
```

Alternatively, install from sources:

```
pip install -e .
```

It is reccomended that installation from source is done inside a virtualenv.

CHAPTER 5

Testing

To confirm that cigram works on your system and the build is functioning use the included tests. These are written in py.test. To run simply install py.test (a requirement of CiGRAM and run).

```
pytest
```

This should run without error.

CHAPTER 6

Acknowledgements

None of this work would have been possible without the help of numerous talented people. This includes the supervision of Jamie Twycross, Andrew Wood, Andrezj Bargeila, Natalio Krasnogor and Michael Holdsworth. Much of this work was also supported by Paweł Widera.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search