Bipartite Partial Configuration Model Documentation

Release 1.0

Mika J. Straka

January 20, 2017

The Bipartite Partial Configuration Model (BiPCM) is a statistical null model for binary bipartite networks. It offers an unbiased method of analyzing node similarities and obtaining statistically validated monopartite projections [Saracco2016].

The BiPCM is related to the Bipartite Configuration Model (BiCM) [Saracco2015], but imposes only constraints on the degrees of one bipartite node layer. It belongs to a series of entropy-based null model for binary biparite networks, see also

- BiCM

- BiRG

Please consult the original articles for details about the underlying methods and applications to user-movie and international trade databases [Saracco2016], [Straka2016].

An example case is illustrated in the *Tutorial*.

HOW TO CITE

If you use the `bipcm` module, please cite its location on Github and the original article [Saracco2016].

1.1 References

GETTING STARTED

2.1 Overview

The `bicm` module is an implementation of the Bipartite Configuration Model (BiCM) as described in the article [Saracco2016]. The BiCM can be used as a statistical null model to analyze the similarity of nodes in undirected bipartite networks. The similarity criterion is based on the number of common neighbors of nodes, which is expressed in terms of Λ-motifs in the original article [Saracco2016]. Subsequently, one can obtain unbiased statistically validated monopartite projections of the original bipartite network.

The construction of the BiCM, just like the related BiPCM and BiRG models, is based on the generation of a grand-canonical ensemble of bipartite graphs subject to certain constraints. The constraints can be of different types. For instance, in the case of the BiCM the average degrees of the nodes of the input network are fixed. In the BiRG, on the other hand, the total number of edges is constrained.

The average graph of the ensemble can be calculated analytically using the entropy-maximization principle and provides a statistical null model, which can be used for establishing statistically significant node similarities. In general, they are referred to as entropy-based null models. For more information and a detailed explanation of the underlying methods, please refer to [Saracco2016].

By using the `bicm` module, the user can obtain the BiCM null model which corresponds to the input matrix representing an undirected bipartite network. To address the question of node similarity, the p-values of the observed numbers of common neighbors can be calculated and used for statistical verification. For an illustration and further details, please refer to [Saracco2016] and [Straka2016].

2.1.1 Dependencies

`bicm` is written in *Python 2.7* and uses the following modules:

- poibin Module for the Poisson Binomial probability distribution
- scipy
- numpy
- multiprocessing
- ctypes
- doctest For unit testing

2.2 BiPCM Quickstart

If you want to get started right away, go ahead and follow the summary below. The `bipcm` module encompasses essentially two steps for the analysis of node similarities in bipartite networks:

1. given an input matrix, create the biadjacency matrix of the BiPCM null model

2. perform a statistical validation of the similarities of nodes in the same layer

The validated node similarities can be used to obtain an unbiased monopartite projection of the bipartite network, as illustrated in [Saracco2016].

For more detailed explanations of the methods, please refer to [Saracco2016], the *Tutorial* and the *API*.

2.2.1 Calculating the p-values of the node similarities

Be `mat` a two-dimensional binary NumPy array, which describes the biadjacency matrix of an undirected bipartite network. The nodes of the two bipartite layers are ordered along the columns and rows, respectively. In the algorithm, the two layers are identified by the boolean values `True` for the **row-nodes** and `False` for the **column-nodes**.

Import the module and initialize the Bipartite Partial Configuration Model:

```
>>> from src.bipcm import BiPCM
>>> cma = BiPCM(bin_mat=mat, constraint=<bool>)
```

The parameter `constraint` specifies whether the degrees of the row-nodes (`constraint = True`) or the degrees of the column-nodes (`constraint = False`) should be constrained.

In order to analyze the similarity of the row-layer nodes and to save the p-values of the corresponding Λ-motifs, i.e. of the number of shared neighbors [Saracco2016], use:

```
>>> cma.lambda_motifs_main(bip_set=True, filename=<filename>)
```

For the column-layer nodes, use:

```
>>> cma.lambda_motifs_main(bip_set=False, filename=<filename>)
```

`bip_set` selects the bipartite node set for which the p-values should be calculated and saved. The filename *<filename>* should contain a relative path declaration. The default name of the output file is *pval_constr_<constraint>_proj_<bip_set>.csv*, where *<contraint>* and *<bip_set>* are either *rows* or *columns* depending on the degree constraint and the parameter choice in `lambda_motifs_main`. By default, the values in the file are separated by tabs, which can be changed using the `delim` keyword.

Subsequently, the p-values can be used to perform a multiple hypotheses testing and to obtain statistically validated monopartite projections [Saracco2016].

If the p-values should not be saved but returned by `lambda_motifs_main`, use:

```
>>> cma.lambda_motifs_main(bip_set=True, write=False)
```

2.3 Tutorial

The tutorial will take you step by step from the biadjacency matrix of a real-data network to the calculation of the p-values. Our example bipartite network will be the following:
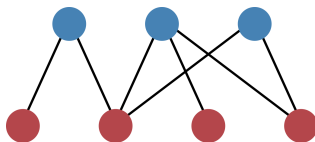
The structure of the network can be caught in the biadjacency matrix. In our case, the matrix is

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

Note that the nodes of the layers of the bipartite network are ordered along the rows and the columns, respectively. In the algorithms, the two layers are identified by the boolean values `True` for the **row-nodes** and `False` for the **column-nodes**. In our example image, the row-nodes are colored in blue (top layer) and the column-nodes in red (bottom layer).

2.4 Testing

The methods in the `bipcm` module have been implemented using doctests. To run the tests, execute:

```
>>> python -m doctest bipcm_tests.txt
```

from the folder *src* in the command line. If you want to run the tests in verbose mode, use:

```
>>> python -m doctest -v bipcm_tests.txt
```

Note that *bipcm.py* and *bipcm_tests.txt* have to be in the same directory to run the test.

2.5 API

API for the methods in the `bipcm` module.

2.6 License

MIT License

Copyright (c) 2015-2017 Mika J. Straka

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

2.7 Contact

For questions or input, please write to mika.straka@imtlucca.it.

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*

[Saracco2015] F. Saracco, R. Di Clemente, A. Gabrielli, T. Squartini, Randomizing bipartite networks: the case of the World Trade Web, Scientific Reports 5, 10595 (2015)

[Saracco2016] F. Saracco, M. J. Straka, R. Di Clemente, A. Gabrielli, G. Caldarelli, T. Squartini, Inferring monopartite projections of bipartite networks: an entropy-based approach, arXiv preprint arXiv:1607.02481

[Straka2016] M. J. Straka, F. Saracco, G. Caldarelli, Product Similarities in International Trade from Entropy-based Null Models, Complex Networks 2016, 130-132 (11 2016), ISBN 978-2-9557050-1-8