The Journal of Open Source Software

# PyGModels: A Python package for exploring Probabilistic Graphical Models with Graph Theoretical Structures

**Doğu Kaan ERASLAN**[1]

**1** École Pratique des Hautes Études, Université PSL, Paris, France

**DOI:** 10.21105/joss.03115

**Software**

- Review
- Repository
- Archive

**Editor:** Dan Foreman-Mackey

**Reviewers:**

- @eigenfoo
- @ankurankan

**Submitted:** 03 February 2021
**Published:** 13 May 2021



## Summary

Probabilistic Graphical Models (PGMs) are a marriage between "graphs" from graph theory and "probability" from statistics and probability theory. While PGMs are widely used in many fields, we noticed that most existing PGM libraries are implement in a way that doesn't take full advantage of their graphical nature. `PyGModels`' value proposition is that it faithfully implements the graphical nature of PGMs, thereby giving `PyGModels`' instantiated objects both graph-theoretical and statistical properties, which allows users to explore and test inference algorithms that are rooted in graph theory or statistics. `PyGModels` also implements several algorithms of interest on Lauritzen-Wermuth-Frydenberg (LWF) chain graphs, also known as mixed graphs.

## Statement of Need

Though the students of computer science or statistics might find pedagogical value going through source code along with a textbook on probabilistic graphical models (for example Cowell, 2005; Koller & Friedman, 2009; Sucar, 2015), `PyGModels` is mainly targeted at researchers. Let us demonstrate the need for `PyGModels` with a use case. Given a set of categorical random variables in the form of a function specified by a probability distribution, a set of edges that encode a certain independence assumption over these random variables, and a set of factors that factorizes this probability distribution over the graph, `PyGModels` is designed with the following use cases in mind:

1. computation of the posterior probability distribution or most probable explanation conditioned on evidence, and
2. the development of new inference algorithms.

The real forte of `PyGModels` is its support for implementing new algorithms due to its lightweight nature and its direct implementation of statistical and graph theoretic features in the same base class. We mostly follow Koller & Friedman (2009) for statistical conventions, definitions, and inference algorithms. For graph theoretic conventions, we follow Diestel (2017), with algorithms from Erciyes (2018) and Even & Even (2012). Throughout the code, exact pages for algorithmic references are cited in the docstrings for relevant functions.

If the independence assumptions over the random variables requires the graph to be a LWF chain graph where the graph can have both directed and undirected edges, `PyGModels` can

also (a) decompose the chain graph into chain components, (b) moralize the chain graph into a Markov Network, and (c) decompose the chain graph into Conditional Random Fields.

The entire library depends only on Python standard library which makes it easily extensible, and straightforward to integrate or adapt to other projects. Through its rigorous adoption of mathematical definitions of involved concepts, it becomes feasible to extend arbitrary factors through their pointwise product, or apply common graph analysis algorithms such as finding articulation points or bridges, or finding an optimal path defined by a cost function.

## Applications and Similar Works

PGMs are known for their wide range of applications in computer vision, information retrieval, disease diagnosis and more recently, in the context of this author's PhD thesis, annotations of ancient documents.

Other open source Python libraries implementing PGMs include:

- `pyGM` (see Ihler, 2020)
- `pgmPy` (see Indap, 2013)
- `pgm` (see Rauber, 2019)
- `pgmpy` (see Ankan & Panda, 2015b)
- `pyfac` (see Lester, 2016)
- `pomegranate` (see Schreiber, 2018)

The most popular of these are `pgmpy` and `pomegranate`, both of which have been used in several publications (see Ankan & Panda, 2015b, 2015a; Schreiber, 2018). Their functionalities are covered with nice test suites as well. Overall both of them are reliable libraries for using PGMs in production.

`pyGM` and `pgm` are particularly well organized alternatives to `PyGModels`, with `pyGM` being slightly more reliable than `pgm` due to its test suite. `pyfac` is primarily focused on inference over factor graphs and `pgmPy`'s development is inactive (last commit dates to 2013). We will make a small comparison with `pgmpy` most of our remarks hold for other alternatives as well.

`PyGModels` distinguishes from `pgmpy` by its lightweight nature (`PyGModels` depends only on python 3.6 standard library). Our test suite cites its source for most of the expected values in the function docstrings. Factors are specified by a set of random variables and a function whose domain is the Cartesian product of codomains of random variables. In all of the libraries above, a factor is specified through an array of values. This has no direct implications on the output. However, it has implications on the evaluation order of operations. Our implementation is lazier and it conforms to the definition provided by Koller & Friedman (see Koller & Friedman, 2009, pp. 106–107). The last aspect is also the case for other packages, however `PyGModels` differs from them with respect to the data structure used in the implementation.

Another key feature, is `PyGModels`' support of inference on LWF chain graphs. The theoretical foundations of these graphs are best explained by Lauritzen (1996), and its causal interpretation and common misconceptions are discussed by Lauritzen & Richardson (2002). Inference strategies over chain graphs are best exposed by Cowell (2005), and more recently by Dechter (2019). These are also known as mixed models or partially directed acyclic graphs (see Koller & Friedman, 2009, p. 37). With `PyGModels`, once we have the necessary set of factors, we can simply do inference over chain graphs just as we do over other PGMs like Bayesian Networks and Markov Random Fields. We implement several algorithms of interest for chain graphs such as decomposition of chain graphs to chain components, moralization of chain graphs.

## Acknowledgement

We acknowledge contributions from Nihan Özcan on her help with documentation process. We also acknowledge AOROC Laboratory of EPHE, PSL University for providing us the necessary tooling support during the development phase of this library.

## References

Ankan, A., & Panda, A. (2015a). *Mastering probabilistic graphical models using python: Master probabilistic graphical models by learning through real-world problems and illustrative code examples in python*. ISBN: 978-1-78439-468-4

Ankan, A., & Panda, A. (2015b). Pgmpy: Probabilistic graphical models using python. *Proceedings of the 14th Python in Science Conference (SCIPY 2015)*. https://doi.org/10.25080/Majora-7b98e3ed-001

Cowell, R. G. (2005). *Probabilistic networks and expert systems*. Springer-Verlag. http://accesbib.uqam.ca/cgi-bin/bduqam/transit.pl?&noMan=25126878

Dechter, R. (2019). *Reasoning with probabilistic and deterministic graphical models: Exact algorithms*. https://doi.org/10.2200/S00893ED2V01Y201901AIM041

Diestel, R. (2017). *Graph theory* (5th ed.). Springer. ISBN: 978-3-662-53621-6

Erciyes, K. (2018). *Guide to graph algorithms: Sequential, parallel and distributed* (1st ed. 2018). Springer International Publishing : Imprint: Springer. https://doi.org/10.1007/978-3-319-73235-0

Even, S., & Even, G. (2012). *Graph algorithms* (2nd ed). Cambridge University Press. ISBN: 978-0-521-51718-8

Ihler, A. (2020). *pyGM*. https://github.com/ihler/pyGM

Indap, A. (2013). *pgmPy*. https://github.com/indapa/pgmPy

Koller, D., & Friedman, N. (2009). *Probabilistic graphical models: Principles and techniques*. MIT Press. ISBN: 978-0-262-01319-2

Lauritzen, S. L. (1996). *Graphical models*. Clarendon Press ; Oxford University Press. ISBN: 978-0-19-852219-5

Lauritzen, S. L., & Richardson, T. S. (2002). Chain graph models and their causal interpretations. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *64*(3), 321–348. https://doi.org/10.1111/1467-9868.00340

Lester, R. (2016). *Pyfac*. https://github.com/rdlester/pyfac

Rauber, P. (2019). *Pgm*. https://github.com/paulorauber/pgm

Schreiber, J. (2018). Pomegranate: Fast and flexible probabilistic modeling in python. *arXiv:1711.00137 [cs, Stat]*. http://arxiv.org/abs/1711.00137

Sucar, L. E. (2015). *Probabilistic graphical models: Principles and applications*. Springer. ISBN: 978-1-4471-6698-6