The Journal of Open Source Software

# pyPLNmodels: A Python package to analyse multivariate high-dimensional count data

**Bastien Batardiere** [1¶], **Julien Chiquet**[1], **and Joon Kwon**[1]

**1** Université Paris-Saclay, AgroParisTech, INRAE, UMR MIA Paris-Saclay ¶ Corresponding author

**DOI:** 10.xxxxxx/draft

**Software**

- Review
- Repository
- Archive

**Editor:** Open Journals

**Reviewers:**

- @openjournals

**Submitted:** 01 January 1970
**Published:** unpublished



## Summary

High dimensional count data are hard to analyse as is, and normalization must be performed but standard normalization does not fit to the characteristics of count data. The Poisson LogNormal(PLN) (Aitchison & Ho, 1989) and PLN-PCA (Chiquet et al., 2018) are two-sided models allowing both suitable normalization and analysis of multivariate count data. They are both implemented in the `pyPLNmodels` package introduced here. Possible fields of applications include

- Ecology: for $n$ sites and $p$ species, the counts represents the number of individuals of each species in each site. The PLN models aims to understand the correlation between species, specifically to establish potential dependencies, competitive interactions, and predatory dynamics. Additionally, the PLN model seeks to explain the impact of covariates, such as temperature, altitude, and other relevant factors, on the observed abundances.
- Genomics: for $n$ cells and $p$ genes, the counts represents the number of times a gene is expressed in each cell. The objective is to estimate the correlation between genes and reduce the number of variables.

The models can deal with offsets when needed. The main functionalities of the `pyPLNmodels` package are

- Normalize count data to obtain more valuable data
- Analyse the significance of each variable and their correlation
- Perform regression when covariates are available
- Reduce the number of variables with the PLN-PCA model
- Visualize the normalized data

To illustrate the main model's effect, we display below a visualization of the first two principal components when Principal Component Analysis (PCA) is performed with the PLN-PCA model on the left and standard PCA on the log normalized data on the right. The data considered is the `scMARK` benchmark (Diaz-Mejia, 2021) described in the benchmark section. We kept 1000 samples for illustration purposes.

The Journal of Open Source Software

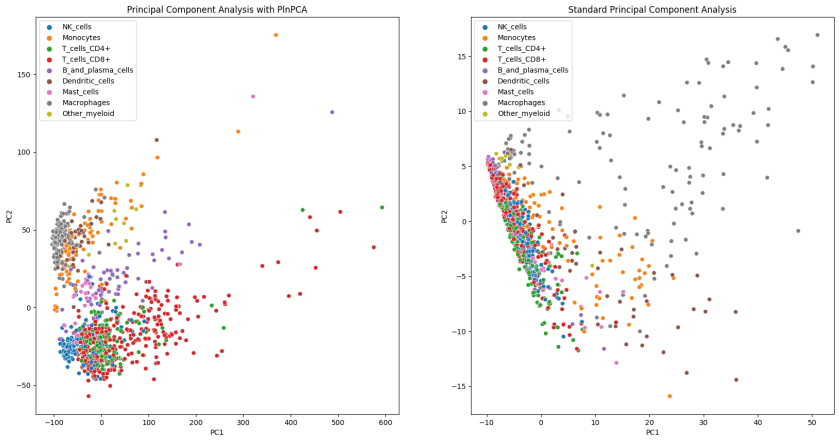


**Figure 1:** PLN-PCA on the left and standard PCA on the right.

## Statement of need

While the R-package `PLNmodels` (Chiquet et al., 2021) implements PLN models, the python package `pyPLNmodels` based on Pytorch (Paszke et al., 2019) has been built to handle large datasets of count data, such as scRNA (single-cell Ribonucleic acid) data. Real-world scRNA datasets typically involves thousands of cells ($n \approx 20000$) with thousand of genes($p \approx 20000$), resulting in a matrix of size $\approx 20000 \times 20000$. The package has GPU support for a better scalability.

The `statsmodels` (Seabold & Perktold, 2010) python package allows to deal with count data thanks to the Generalized Linear Models `PoissonBayesMixedGLM` and `BinomialBayesMixedGLM` classes. We stand out from this package by allowing covariance between variables and performing Principal Component Analysis adequate to count data.

## Benchmark

We fit PLN and PLN-PCA models with the `pyPLNmodels` package on the `scMARK` dataset, a benchmark for scRNA (sincle-cell Ribnonucleic acid ) data with $n = 19998$ samples (cells) and 14059 features (gene expression). The dataset gives access to the cell type of each sample, taking 28 different values. We plot below the running times required to fit such models when the number of variables (i.e. genes) grows in FIG. We used 60 Principal Components when fitting the PLN-PCA model. A tolerance must be set as stopping criterion when fitting each model. The running time required with the default is plot in solid line and a dotted line is plot with a relaxed tolerance. Note that the default tolerance ensures the model parameters have reached convergence but the relaxed one gives satisfying model parameters, while being much faster.

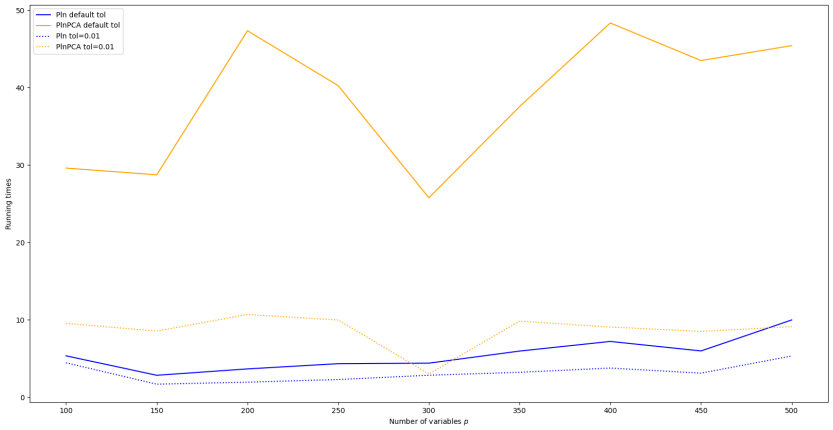


**Figure 2:** Running time analysis on the scMARK benchmark.


## Acknowledgements

The authors would like to thank Jean-Benoist Léger for the time spent on giving precious advices to build a proper python package. This work was supported by the French ANR SingleStatOmics.


## References


Aitchison, J., & Ho, C. H. (1989). The multivariate Poisson-log normal distribution. *Biometrika*, *76*(4), 643–653. https://doi.org/10.1093/biomet/76.4.643

Chiquet, J., Mariadassou, M., & Robin, S. (2018). VARIATIONAL INFERENCE FOR PROBABILISTIC POISSON PCA. *The Annals of Applied Statistics*, *12*(4), 2674–2698. https://www.jstor.org/stable/26666168

Chiquet, J., Mariadassou, M., & Robin, S. (2021). The poisson-lognormal model as a versatile framework for the joint analysis of species abundances. *Frontiers in Ecology and Evolution*. https://doi.org/10.3389/fevo.2021.588292

Diaz-Mejia, J. (2021). *scMARK an 'MNIST' like benchmark to evaluate and optimize models for unifying scRNA data* (Version 1.0) [Data set]. Zenodo. https://doi.org/10.5281/zenodo.5765804

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., … Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems 32* (pp. 8024–8035). Curran Associates, Inc. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Seabold, S., & Perktold, J. (2010). Statsmodels: Econometric and statistical modeling with python. *9th Python in Science Conference*.