




libscientific: A Powerful C Library for Multivariate Analysis

Giuseppe Marco Randazzo ¹

¹ Independent researcher

DOI: [10.21105/joss.05420](https://doi.org/10.21105/joss.05420)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Mehmet Hakan Satman](#) 



Reviewers:

- [@mikeaalv](#)
- [@faosorios](#)

Submitted: 13 March 2023

Published: 25 October 2023

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

Summary

Multivariate analysis is a powerful technique that allows researchers to analyze and interpret data with multiple variables. In today's data-driven world, multivariate analysis has become essential for the exploration of complex data sets. libscientific is a powerful library written in C that provides a comprehensive set of multivariate analysis tools based on the NIPALS algorithm. The library includes several multivariate analysis algorithms, such as principal component analysis (PCA), partial least squares regression (PLS), consensus principal component analysis (CPCA), multiblock principal component analysis, and multiblock partial least squares (UPLS). libscientific also includes several other tools to analyze data, such as cluster analysis using KMeans Hierarchical clustering and other methods to run linear algebra calculations. The library also provides a Python foreign function to be used inside Python scripts.

Statement of need

The library is designed to be easy to use and can be integrated into any C or C++ project. Additionally, libscientific comes with a foreign function Python bindings, making it accessible within Python scripts and easier to perform data analysis tasks. One of the main advantages of libscientific is its performance and scalability. This means that large data sets can be analyzed quickly and efficiently, making it an ideal choice for applications where speed is critical. The library depends only on lapack for SVD and eigenvalues decomposition and can be easily integrated into embedded systems. The current library version is 1.6.0, and here is a list of the current library features:

- Principal Component Analysis (PCA)
- Consensus Principal Component Analysis (CPCA)
- Partial Least Squares (PLS)
- Multiple Linear Regression (MLR)
- Unfold Principal Component Analysis (UPCA)
- Unfold Partial Least Squares (UPLS)
- Fisher Linear Discriminant Analysis (LDA)
- Kmeans++ Clustering
- Hierarchical Clustering
- Sample selection algorithms: Most Descriptive Compound (MDC), Most Dissimilar Compound (MaxDis)
- Statistical measures: R2, MSE, MAE, RMSE, Sensitivity, PPV
- Yates Analysis
- Receiver Operating Characteristic curve analysis (ROC)
- Precision-Recal curve analysis
- Matrix-matrix Euclidean, Manhattan, Cosine and Mahalanobis distances
- Numerical integration

- Natural cubic spline interpolation and prediction
- Linear algebra (Eigenvector/value and SVD operated by Lapack library)
- Ordinary Least Squares solver
- Linear equation Solver
- Nelder-Mead Simplex Optimization
- Cross validation methods: Bootstrap k-fold, Leave-One-Out, Y-Scrambling

libscientific was designed to analyze any kind of multivariate tabular data and to be applied in any scientific field.

State of The field

The primary objective of libscientific is to offer a library capable of performing multivariate analysis by implementing the NIPALS algorithm. This choice stems from the limitations of prevailing popular libraries like scikit-learn, which need help handling missing and noisy data effectively (Ke & Kanade, 2005; Little & Rubin, 1987). Notably, the NIPALS algorithm is a robust solution to these challenges, addressing issues related to data incompleteness without necessitating prior data imputation. Furthermore, the NIPALS algorithm conducts iterative computations of components and latent variables, leading to more efficient use of memory resources than alternative methods found in scikit-learn. Notably, it demonstrates superior computational efficiency, especially when researchers seek to analyze only a select few of the foremost principal components or latent variables. In summary, libscientific aims to provide a sophisticated solution for multivariate analysis, leveraging the NIPALS algorithm's strengths to surmount issues related to missing and noisy data, optimize component calculations, and enhance computational efficiency in scenarios where the analysis focuses on a limited number of critical principal components or latent variables.

Multivariate analysis algorithms specs

Principal component analysis (PCA) is one of the most commonly used methods for multivariate analysis. PCA is an unsupervised method that compresses data into low-dimensional representations that capture the dominant variation in the data. libscientific provides a robust implementation of PCA using the NIPALS algorithm described in Geladi & Kowalski (1986). libscientific implementation can handle data sets with many variables, few instances, and missing values.

Partial least squares (PLS) is another commonly used method for multivariate analysis. PLS is a supervised method that captures the dominant covariation between the data matrix and the target/response. libscientific provides one version of PLS described by Geladi & Kowalski (1986). This implementation works with single-task and multi-task regression problems.

In addition to PCA and PLS, libscientific provides implementations of Consensus PCA (CPCA) to analyze time series and multi-block data, algorithm described by Westerhuis et al. (1998), and other multi-block methods such as Unfold Principal Component Analysis (UPCA) and Unfold Partial Least Squares (UPLS) both implementation from Wold et al. (1987).

All multivariate algorithms admit missing values since the core linear algebra functions are coded to skip missing values, according to Martens & Martens (2001), p. 381.

Other algorithms

The library also provides compound selection algorithms such as Most Descriptive Compounds (Hudson et al., 1996) or Most Dissimilar Compound (Holliday & Willett, 1996) selections, allowing one to analyze scores plots or original data matrices and select samples based on the object/sample diversity. Moreover, multi-thread cross-validation methodologies such as

“Bootstrap k-fold” Leave-One-Out (LOO), and Y-Scrambling tests are implemented to facilitate the scientist in testing model prediction abilities.

Algorithm stability

Since we are dealing with numerical analysis, unit tests are crucial to ensure correctness, stability, and reproducibility. libscientific tests range from simple matrix-vector multiplication to the correctness of complex algorithms using ad-hoc torture toy examples. The code coverage is reported to be more than 75%, indicating that a larger portion of the code has been verified to work as expected, reducing the likelihood of undiscovered bugs. This is important since libscientific is a set of implementations of algorithms that involves complex mathematical calculation, and correctness and accuracy are crucial in minimizing the risk of numerical errors in scientific, engineering, and data analysis applications.

Speed and Memory Comparison

Several simulations of every algorithm in libscientific with data of different sizes (input size) against CPU speed were performed to address the algorithm’s performance.

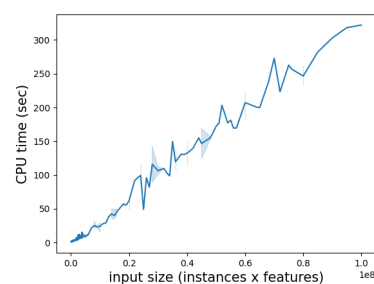
Looking at the plots for PCA, CPCA, and PLS in Figure 1, we observe a linear trend, which suggests that the algorithm’s time complexity also increases linearly. However, this does not tell the computational complexity of the algorithms. Indeed, since the NIPALS algorithm is similar to the power method for determining the eigenvectors and eigenvalues of a matrix (Lorber et al., 1987), we can assume that the computational complexity could be $O(n^2)$ or $O(n^3)$. Moreover it is important to point out that the calculation speed is mainly influenced by the number of samples, the number of iterations required for convergence, and the number of principal components/latent variables to calculate.

Instead, MLR shows a polynomial correlation as expected from the OLS algorithm, which uses a matrix direct inverse approach with a computational complexity of $O(n^3)$.

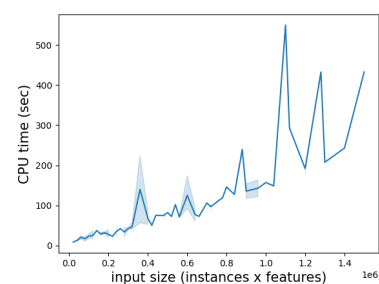
With this analysis, we confirm that as the input size (often termed “problem size”) increases by a constant factor, the execution time also increases proportionally (linear algorithms). Linear algorithms have notable characteristics:

- Most of the time, ‘Linear Time Complexity ($O(n)$)’: Execution time grows linearly with input size.
- Constant Work per Input Element: Each input element is processed continuously in linear algorithms.
- Stable Performance Impact: Doubling input size roughly doubles execution time, facilitating performance estimation.
- Optimal Scaling: Linear-time solutions efficiently handle larger inputs.

PCA



CPCA



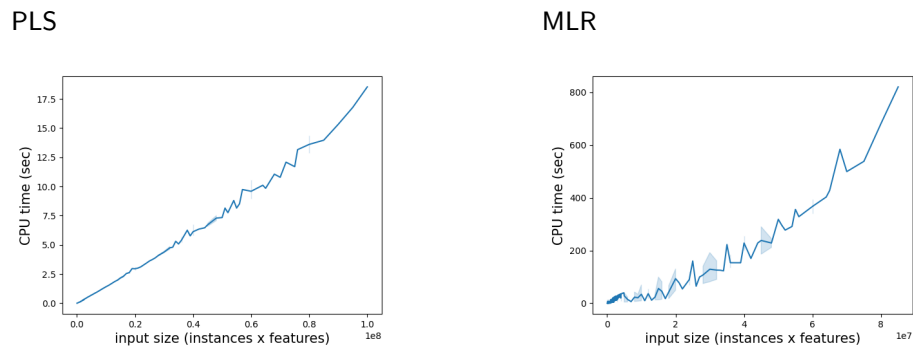


Figure 1: Speed performances of 4 different algorithms: Principal Component Analysis (PCA), Partial Least-Squares (PLS), Consensus Principal Component Analysis (CPCA), and Multiple Linear Regression (MLR). Simulations reveal linear trends for PCA, CPCA, and PLS, hinting at probable linear time complexity. However, it is worth mentioning that the NIPALS algorithm, influenced by sample size, iterations, latent variables, and similar to the power method for estimating eigenvectors/eigenvalues, may report $O(n^2)$ or $O(n^3)$ complexity. Meanwhile, MLR exhibits a polynomial correlation typical of the OLS matrix approach with $O(n^3)$ complexity.

Usage

For the usage in C or either Python we invite reading the official documentation located at the following link: <http://gmrando.github.io/libscientific/>

Conclusions

libscientific offers a potent suite of multivariate analysis tools that greatly enhance the ability of researchers and analysts to extract valuable insights from diverse tabular data. With its robust C-based implementation and seamless Python bindings, the library balances high performance and user-friendliness, making it an optimal solution for swiftly executing data-driven applications.

Incorporating libscientific into analytical workflows may empower professionals to leverage various multivariate techniques to crack complex relationships and patterns within datasets. By offering tools for data reduction, predictive modeling, quality control, and more, as already demonstrated in previous works in -omics science and predictive modeling (Kwon et al., 2021, 2022; Randazzo et al., 2016, 2020; Randazzo, Vigneau, et al., 2017; Randazzo, Tonoli, et al., 2017), the library can be an indispensable asset for tackling intricate challenges across various disciplines.

Acknowledgements

libscientific was born as an open-source project from the Ph.D. thesis of the author Giuseppe Marco Randazzo. The author acknowledges the support from the University of Perugia, the valuable code review made by the people from Freaknet Medialab, and the bug reports from the whole open-source community using this library.

References

Geladi, P., & Kowalski, B. R. (1986). Partial least-squares regression: A tutorial. *Analytica Chimica Acta*, 185, 1–17. [https://doi.org/10.1016/0003-2670\(86\)80028-9](https://doi.org/10.1016/0003-2670(86)80028-9)

- Holliday, J. D., & Willett, P. (1996). Definitions of "dissimilarity" for dissimilarity-based compound selection. *SLAS Discovery*, 1(3), 145–151. <https://doi.org/10.1177/108705719600100308>
- Hudson, B. D., Hyde, R. M., Rahr, E., Wood, J., & Osman, J. (1996). Parameter based methods for compound selection from chemical databases. *Quantitative Structure-Activity Relationships*, 15(4), 285–289. <https://doi.org/10.1002/qsar.19960150402>
- Ke, Q., & Kanade, T. (2005). *Robust L1 norm factorization in the presence of outliers and missing data by alternative convex programming*. 1, 739–746 vol. 1. <https://doi.org/10.1109/CVPR.2005.309>
- Kwon, H.-K., Dussik, C. M., Kim, S.-H., Kyriakides, T. R., Oh, I., & Lee, F. Y. (2022). Treating "septic" with enhanced antibiotics and "arthritis" by mitigation of excessive inflammation. *Frontiers in Cellular and Infection Microbiology*, 12. <https://doi.org/10.3389/fcimb.2022.897291>
- Kwon, H.-K., Lee, I., Yu, K. E., Cahill, S. V., Alder, K. D., Lee, S., Dussik, C. M., Back, J., Choi, J., Song, L., Kyriakides, T. R., & Lee, F. Y. (2021). Dual therapeutic targeting of intra-articular inflammation and intracellular bacteria enhances chondroprotection in septic arthritis. *Science Advances*, 7(26), eabf2665. <https://doi.org/10.1126/sciadv.abf2665>
- Little, R. J. A., & Rubin, D. B. (1987). *Statistical analysis with missing data*. Wiley. <https://doi.org/10.1002/9781119013563>
- Lorber, A., Wangen, L. E., & Kowalski, B. R. (1987). A theoretical foundation for the PLS algorithm. *Journal of Chemometrics*, 1(1), 19–31. <https://doi.org/10.1002/cem.1180010105>
- Martens, H., & Martens, M. (2001). *Multivariate analysis of quality : An introduction*. Wiley. <https://doi.org/10.1088/0957-0233/12/10/708>
- Randazzo, G. M., Bileck, A., Danani, A., Vogt, B., & Groessl, M. (2020). Steroid identification via deep learning retention time predictions and two-dimensional gas chromatography-high resolution mass spectrometry. *Journal of Chromatography A*, 1612, 460661. <https://doi.org/10.1016/j.chroma.2019.460661>
- Randazzo, G. M., Tonoli, D., Hambye, S., Guillarme, D., Jeanneret, F., Nurisso, A., Goracci, L., Boccard, J., & Rudaz, S. (2016). Prediction of retention time in reversed-phase liquid chromatography as a tool for steroid identification. *Analytica Chimica Acta*, 916, 8–16. <https://doi.org/10.1016/j.aca.2016.02.014>
- Randazzo, G. M., Tonoli, D., Strajhar, P., Xenarios, I., Odermatt, A., Boccard, J., & Rudaz, S. (2017). Enhanced metabolite annotation via dynamic retention time prediction: Steroidogenesis alterations as a case study. *Journal of Chromatography B*, 1071, 11–18. <https://doi.org/10.1016/j.jchromb.2017.04.032>
- Randazzo, G. M., Vigneau, E., Courcoux, P., Harrouet, C., Lijour, Y., Dardenne, P., Boccard, J., & Rudaz, S. (2017). Indirect quantitative structure-retention relationship for steroid identification: A chemometric challenge at "chimiométrie 2016." *Chemometrics and Intelligent Laboratory Systems*, 160, 52–58. <https://doi.org/10.1016/j.chemolab.2016.11.010>
- Westerhuis, J. A., Kourti, T., & MacGregor, J. F. (1998). Analysis of multiblock and hierarchical PCA and PLS models. *Journal of Chemometrics*, 12(5), 301–321. [https://doi.org/10.1002/\(SICI\)1099-128X\(199809/10\)12:5%3C301::AID-CEM515%3E3.0.CO;2-S](https://doi.org/10.1002/(SICI)1099-128X(199809/10)12:5%3C301::AID-CEM515%3E3.0.CO;2-S)
- Wold, S., Geladi, P., Esbensen, K., & Öhman, J. (1987). Multi-way principal components- and PLS-analysis. *Journal of Chemometrics*, 1(1), 41–56. <https://doi.org/10.1002/cem.1180010107>