The Journal of Open Source Software

PyArmadillo: an alternative approach to linear algebra in Python

Jason Rumengan[1, 2], **Terry Yue Zhuo**[3], **and Conrad Sanderson**[1, 4]

1 Data61/CSIRO, Australia **2** Queensland University of Technology, Australia **3** University of New South Wales, Australia **4** Griffith University, Australia

DOI: 10.21105/joss.03051

Software

- Review
- Repository
- Archive

Editor: Matthew Sottile

Reviewers:

- @JaroslavHron
- @uellue

Submitted: 10 February 2021
Published: 22 February 2021

Summary

PyArmadillo is a streamlined linear algebra library for the Python language, with an emphasis on ease of use. It aims to provide a high-level syntax and functionality deliberately similar to Matlab/Octave (Linge & Langtangen, 2016), allowing mathematical operations to be expressed in a familiar and natural manner. PyArmadillo provides objects for matrices and cubes, as well as over 200 associated functions for manipulating data stored in the objects. Integer, floating point and complex numbers are supported. Various matrix factorisations are provided through integration with LAPACK (Anderson et al., 1999), or one of its high performance drop-in replacements such as Intel MKL (Intel, 2016) or OpenBLAS (Zhang et al., 2016).

Statement of need

While frameworks such as NumPy (Harris et al., 2020) and SciPy (Virtanen et al., 2020) are available for Python, they tend to be overly verbose and somewhat cumbersome to use in comparison to Matlab/Octave. From a linear algebra point of view, the issues include: **(i)** a focus on operations involving multi-dimensional arrays rather than matrices, **(ii)** nested organisation of functions which increases code verbosity (which in turn can lead to reduced user productivity), and **(iii)** syntax that notably differs from Matlab. PyArmadillo aims to address these issues by primarily focusing on linear algebra, having all functions accessible in one flat structure, and providing an API that closely follows Matlab.

Implementation

PyArmadillo relies on Armadillo (Sanderson & Curtin, 2018, 2016) for the underlying C++ implementation of matrix objects and associated functions, as well as on pybind11 (Jakob et al., 2017) for interfacing C++ and Python. Due to its expressiveness and relatively straightforward use, pybind11 was selected over other interfacing approaches such as Boost.Python (Abrahams & Grosse-Kunstleve, 2003) and manually writing C++ extensions for Python. Armadillo was selected due to its API having close similarity to Matlab, as well as previous success with RcppArmadillo, a widely used bridge between the R language and C++ (Eddelbuettel & Sanderson, 2014).

Acknowledgements

We would like to thank our colleagues at Data61/CSIRO (Andrew Bolt, Dan Pagendam, Piotr Szul) for providing feedback and initial testing.

References

Abrahams, D., & Grosse-Kunstleve, R. W. (2003). Building hybrid systems with Boost.Python. *C/C++ Users Journal*, *21*(7). https://www.osti.gov/biblio/815409

Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., & Sorensen, D. (1999). *LAPACK users' guide*. Society for Industrial and Applied Mathematics. https://doi.org/10.1137/1.9780898719604

Eddelbuettel, D., & Sanderson, C. (2014). RcppArmadillo: Accelerating R with high-performance C++ linear algebra. *Computational Statistics & Data Analysis*, *71*, 1054–1063. https://doi.org/10.1016/j.csda.2013.02.005

Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., R'ıo, J. F. del, Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Intel. (2016). *Math Kernel Library (MKL)*. http://software.intel.com/en-us/intel-mkl/

Jakob, W., Rhinelander, J., & Moldovan, D. (2017). pybind11 – seamless operability between C++11 and Python. In *GitHub repository*. GitHub. https://github.com/pybind/pybind11

Linge, S., & Langtangen, H. P. (2016). *Programming for computations - MATLAB/Octave*. Springer. https://doi.org/10.1007/978-3-319-32452-4

Sanderson, C., & Curtin, R. (2018). A user-friendly hybrid sparse matrix class in C++. *Lecture Notes in Computer Science (LNCS), Vol. 10931*, 422–430. https://doi.org/10.1007/978-3-319-96418-8_50

Sanderson, C., & Curtin, R. (2016). Armadillo: A template-based C++ library for linear algebra. *Journal of Open Source Software*, *1*, 26. https://doi.org/10.21105/joss.00026

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, *17*, 261–272. https://doi.org/10.1038/s41592-019-0686-2

Zhang, X., Wang, Q., & Saar, W. (2016). OpenBLAS: An optimized BLAS library. In *GitHub repository*. GitHub. https://github.com/xianyi/openblas