Email address

mhaberla@calpoly.edu

Name of Submitter

Matt Haberland

Project Submitting this Proposal

SciPy

Is your project Affiliated or Sponsored?

Sponsored

Proposal Title

Enhanced LAPACK Support in SciPy

Two Sentence Summary of Proposal

Scientists and engineers worldwide rely on SciPy for linear algebra computations, and SciPy in turn relies on the Fortran library LAPACK (Linear Algebra PACKage); however, SciPy currently lacks support for many important LAPACK functions. Our proposal is to remedy this by writing Python wrappers for these missing LAPACK subroutines.

Description of Proposal (<750 words, < 4500 chars)

LAPACK (Linear Algebra PACKage [1]) is a standard, permissively-licensed software library that provides functions for performing matrix factorizations, solving systems of linear equations, and solving eigenvalue problems. Written in Fortran and optimized for performance, LAPACK is the engine SciPy uses to enable fast numerical linear algebra computations from Python, a convenient but comparatively slow language. As recognized in SciPy's guiding document, the SciPy Roadmap [2], the Python interface to LAPACK is "one of the most important things that SciPy Provides"; indeed, there are nearly 50,000 code results on GitHub that import `scipy.linalg.lapack` [3, 4]. SciPy currently includes Python wrappers (Python-accessible versions) of 461 LAPACK functions, but for every one of these, two are missing. With the aid of a Small Development Grant, we aim to make SciPy's coverage of LAPACK more comprehensive by adding wrappers for 68 of the most important missing subroutines.

We took three approaches to prioritize LAPACK functions for inclusion in this project: studying open issues involving LAPACK on SciPy's GitHub repository, categorizing LAPACK routines and selecting those in the most fundamental categories, and analyzing usage of LAPACK routines on GitHub. The following list is a summary of the LAPACK functions we will add. (Note that as there are four common numerical data types used in computational linear algebra – single precision floating point, double precision floating point, single precision complex, and double precision complex – there are four variants of many LAPACK subroutines, each prefixed with a letter "s", "d", "c", or "z" to identify associated data type. For brevity, we use "?" as a wildcard character to refer to one or all of these prefixes, and thus each item in

the list refers to between one and four LAPACK subroutines that differ only in their associated data type. All LAPACK subroutines are documented at [5].)

?getc2 / ?gesc2 (8 subroutines) – ?getc2 computes the LU factorization of a matrix with complete pivoting; ?gesc2 solves a linear system using that factorization. Requested in GitHub issue #10788; unmerged PR #10797 will be completed as part of this project.

?syevr / ?heevr (4 subroutines) – computes an eigenvalue decomposition of a real symmetric / Hermitian matrix. Currently implement with reduced functionality; exposure of additional features requested in GitHub issues #6502. Unmerged PR #6510 will be completed as part of this project.

?gejsv (4 subroutines) – computes the singular value decomposition of a general matrix with the advantage of higher accuracy computation of tiny singular values and their singular vectors compared to other LAPACK SVD functions. Requested in GitHub issue #5616.

?pttrf / ?pttrs / ?pteqr (12 subroutines) – ?pttrf computes the LDLT factorization of a real symmetric positive definite (SPD) tridiagonal matrix, ?pttrs solves a linear system using that factorization, and ?pteqr computes the eigenvalues and eigenvectors of a SPD tridiagonal matrix. SciPy has wrappers tailored for the solution of systems and eigenvalue problems for other matrix types, and SPD tridiagonal matrices are important matrix class, so SciPy would benefit from enhanced support for SPD tridiagonal matrices.

?gttrf / ?gttrs (8 subroutines) – ?gttrf computes the LU factorization of a real tridiagonal matrix, and ?gttrs solves a linear system using that factorization. The argument for including these is similar to that for the routines above.

?geqrfp (4 subroutines) – computes the QR decomposition of a real general matrix with non-negative elements on the diagonal of R. This is an important variant of QR for some machine learning calculations [7, 8].

?tbtrs (4 subroutines) – solves a linear system with a triangular banded matrix. Triangular band matrices are an important class of matrices, and solving such systems using a tailored routine can yield substantial speed improvements.

?gtsvx, ?ptsvx, ?gbsvx, ?pbsvx (16 subroutines) – solves a linear system with general tridiagonal, SPD tridiagonal, general banded, and SPD banded matrices. Importantly, these routines also provide error bounds on the solution and a condition estimate. SciPy already includes wrappers for the equivalent routines for other matrix types; these would complete the set.

?gesvdx (4 subroutines) – computes the singular value decomposition of a general matrix or, optionally, a subset of its singular value and vectors. Requested in GitHub issue #8231 [6].

?ggsvd3 (4 subroutines) – computes the generalized singular value decomposition of two matrices. Requested in GitHub issues #743 and #1491.

References:

[1] Anderson, Edward, et al. "LAPACK: A portable linear algebra library for high-performance computers." *Proceedings of the 1990 ACM/IEEE conference on Supercomputing*. IEEE Computer Society Press, 1990.

[2] The SciPy Community. "SciPy Roadmap." *SciPy v1.3.1 Reference Guide,* https://docs.scipy.org/doc/scipy/reference/roadmap.html. Accessed 15 October 2019.

[3] Query results for "scipy.linalg.lapack". *GitHub.* https://github.com/search?l=Python&q="scipy.linalg.lapack"&type=Code. Accessed 15 October 2019.

[4] Query results for "from scipy.linalg import lapack". *GitHub.* https://github.com/search?l=Python&q=%22from+scipy.linalg+import+lapack%22&type=Code. Accessed 15 October 2019.

[5] "LAPACK". *Netlib Repository*. http://www.netlib.org/lapack/explore-html/d8/d70/group__lapack.html. Accessed 19 October 2019.

[6] The SciPy Community. "GitHub Issues". *GitHub*. https://github.com/scipy/scipy/issues/. Accessed 19 October 2019.

[7] @markusm73. "QR factorization still producing negative diagonals?". *LAPACK/ScaLAPACK Development*. https://icl.cs.utk.edu/lapack-forum/viewtopic.php?f=2&t=5213. Accessed 19 October 2019.

[8] James W. Demmel et al. "Nonnegative Diagonals and High Performance on Low-Profile Matrices from Householder QR". *SIAM Journal on Scientific Computing.* https://www.researchgate.net/publication/220411459. Accessed 19 October 2019.

Benefit to Project/Community (<400 words, < 2500 chars)

This project would resolve several GitHub issues and unmerged PRs, some of which have been stagnant for years, and it partially addresses the first item ("Evolve BLAS and LAPACK support") on SciPy's top-level roadmap, which is reserved for "only the most important ideas and needs for SciPy".

Amount Requested

$4,978

Brief Budget Justification - How will the money be spent?

$1067 ($983 salary; $84 fringe benefits/payroll taxes) will compensate the proposer for 15 hours of overload work, approximately one hour per unique wrapper. The proposer's primary responsibility will be to design and implement unit tests.

$1027 will cover 50 hours of an undergraduate student's time ($1,000 wages, $27 fringe benefits/employer payroll taxes), approximately three hours per unique wrapper. The undergraduate's primary responsibility will be to draft the wrappers using F2PY.

$1,500 is for 30 hours of work by Kai Striega at a rate of $50 (USD)/hr. Kai Striega's primary responsibility will be to provide code reviews and support for unforeseen challenges.

$1,384 is for Cal Poly recovery of indirect costs.

The Cal Poly salary and wage rates are based on the California Polytechnic State University (CPSU) and Cal Poly Corporation (CPC), jointly Cal Poly, established salary and wage rates paid during the 2019-2020 Fiscal year (July 1 – June 30). Benefits for CPSU Faculty summer and overload work include FICA, SUI and Workers Compensation and are calculated at the DHHS pooled rate of 8.5%. CPC undergraduate student benefits include SUI and Worker's Compensation. The DHHS pooled rate of 2.7% is used for budgetary purposes. Only rates in effect at the time the work is performed will be charged to the project. Cal Poly's federally negotiated indirect rate is 38.5% of modified total direct costs, effective July 1, 2019. Modified total direct costs exclude equipment, capital expenditures, charges for patient care, tuition remission, rental costs of off-site facilities, scholarships, and fellowships, participant support costs, and the portion of each subaward in excess of $25,000.

Kai Striega's rate of $50/hr is approximately equivalent to the $80 (AUD) commensurate with his expertise and experience.

Timeline of Deliverables

11/18/2019 – Acceptance notification

The wrappers can be written in any order, so we will begin at the top of the list and complete wrappers and corresponding unit tests at an average rate of three wrappers every two weeks during the three-month project period. Each wrapper and its corresponding unit tests will be submitted in a unique PR. After each PR is generated, we will allow approximately one week for additional feedback from the community before merging.

2/18/2020 – Submit final report

Project Team

Matt Haberland, Kai Striega, Fletcher Easton

I agree to submit a grant report-back if my proposal is selected for funding.

I agree.