Short Narrative Bio: Matt Haberland is an Assistant Professor in the BioResource and Agricultural Engineering Department at Cal Poly, San Luis Obispo. He earned his Ph.D. in Mechanical Engineering at MIT in 2014 for his thesis "Extracting Principles from Biology for Application to Running Robots", and previously, he created the Contact Sensor / Stabilizer for the rock drill of the Mars rover Curiosity. Dr. Haberland has received several honors for teaching, including the "Distinguished Teaching Award" from the UCLA Math department in 2017, and he has published research in academic journals including IEEE Robotics and Automation Letters, Bioinspiration and Biomimetics, Robotica, and Nature Methods.

1. Proposal Title: Direct Sparse Linear Algebra in SciPy

2. Did you previously apply for funding for this or a related proposal under the CZI EOSS program?

Yes. EOSS-0000000432 was also for SciPy, and EOSS3-0000000269 also proposed the addition of sparse linear algebra solvers.

3. Have you previously received funding for this proposal under the CZI EOSS program?

No. EOSS-0000000432 was for work on SciPy's statistics functionality

4. Proposal Purpose

Limit to one sentence (maximum of 255 characters, including spaces)

To better serve biomedical applications, SciPy will enhance its large-scale sparse linear algebra capabilities.

5. Amount Requested - Enter total budget amount requested in USD, including indirect costs; this number should be between $100k and $400k total costs over a two-year period.

Year One: $64,477
Year Two: $66,383
Total All Years: $130,860

6. Proposal Summary/Scope of Work: A short summary of the application (maximum of 500 words)

SciPy is a Python library that provides fundamental building blocks for modeling and solving scientific problems. It is used directly by biomedical researchers worldwide, and it serves as a foundation for other essential biomedical tools. Developed almost entirely by volunteers for its two-decade history, a few gaps have surfaced, and the one we propose to address is the lack of support for direct solution of sparse systems of linear equations.

The solution of linear equations is one of the most fundamental problems in scientific computing. Even though equations in biomedicine and other fields are often nonlinear, algorithms for more general problems (e.g. nonlinear algebraic equations, differential equations, and optimization problems) include the solution of a linear system as a subroutine. SciPy has very good support for linear algebra when the matrices involved are relatively small (e.g. ~1000 columns). Today's problems, however, routinely include orders of magnitude more unknowns, making them intractable to these solvers. Fortunately, as problem sizes grow, they often become sparse: each variable is coupled to only a few others. Sparse linear solvers exploit this structure to achieve much more efficient computation. Although SciPy includes one direct sparse linear solver, SuperLU, it is generally much slower than state-of-the-art solvers, and it cannot solve other sparse linear algebra problems (e.g. Cholesky decomposition, QR factorization).

To address this:

* We will add custom sparse Cholesky and QR decomposition routines to SciPy that can be distributed with SciPy under its permissive BSD license.

* Although their copyleft licenses are incompatible for distribution within SciPy, we (can and) will add interfaces to the state-of-the-art SuiteSparse library's matrix factorization routines and solvers.

* Existing wrappers of SuiteSparse for use in Python are currently difficult to install on Windows machines. We will ensure that the proposed features are easily accessible on all major platforms.

* We will benchmark the performance of these new features against their equivalents in Matlab to demonstrate that SciPy is a competitive, but free and open source, alternative to other popular numerical computing libraries.

In addition, the SciPy repository has ~1600 open PRs and issues. The large number is not necessarily a bad sign, as it is representative of the project's popularity and community engagement. On the other hand, nearly 1100 of these are over a year old, and the number grows each year. To reduce this backlog, approximately one-half of our effort will be dedicated to project maintenance: solving these open issues and reviewing PRs.

The proposed work would be completed by two SciPy core developers. Matt Haberland has been a maintainer for three years and is the author of 80 merged pull requests in scipy.linalg, scipy.stats, and scipy.optimize. Nicholas McKibben is a relatively new member of the core team; his expertise in exposing compiled code from Python would be essential to this project. Nicholas and Matt have collaborated on other SciPy enhancements of similar scope, including a wrapper for the HiGHS linear programming library and wrappers for Boost statistical distributions.

7. *Work Plan: A description of the proposed work the applicants are requesting funding for, including resources the applicants will provide that are not part of the requested funding. For software development related work (e.g., engineering, product design, user research), specify how the work fits into the existing software project roadmap. For community outreach related activities (e.g., sprints, training), specify how these activities will be organized, the target audience, and expected outcomes (maximum of 750 words)*

For at least 15 years, SciPy has vendored the permissively-licensed sparse linear system solver SuperLU [1]. However, if the SuiteSparse [2] sparse linear algebra library is found on the user's computer, SciPy defaults to its faster, more capable, copyleft-licensed UMFPACK [3] solver. That way, users who are limited to permissively-licensed code can still solve sparse linear systems with SciPy, and those who are not subject to this restriction benefit from improved performance.

For at least three years, the Detailed SciPy Roadmap [4] has called for improvements to other scipy.sparse.linalg "dsolve" (Direct Solve) features, especially the addition of sparse Cholesky decomposition. It specifically mentions that the CHOLMOD [5] routine from SuiteSparse should be considered. More recently, this was extended to include the addition of other direct sparse linear algebra features within SciPy and to create wrappers for other features of SuiteSparse. To address this:

* Using the theory presented in SuiteSparse author Tim Davis' book [6], Dr. Haberland will add custom sparse Cholesky and QR decomposition routines to SciPy so that users limited to permissively-licensed code will be able to solve sparse linear systems using these factorizations. Dr. Haberland is well-prepared for the task, with years of experience implementing complex algorithms from research papers and books including methods for linear programming [7, 8], a quadratic assignment problem solver [9], and many functions for statistical distributions and tests [10, 11, 12, 13, 14, 15, 16, 17, 18, 19].

* We will will add interfaces to the SuiteSparse library's Cholesky (CHOLMOD) and QR (SPQR [20]) factorization routines and solvers. When installed, these will be SciPy's default sparse Cholesky and QR features, so that users can take advantage of their improved speed. Dr. Haberland and Mr. McKibben have collaborated on similar interfaces to compiled code before, such as those for the HiGHS linear programming library [21] and statistical distributions from BiasedUrn [22] and Boost [23].

* There are existing Python wrappers of SuiteSparse's features (scikit-umfpack [24], scikit-sparse [25], and PySPQR [26]), but they are not regularly maintained, none support installation on Windows (despite many requests from would-be users), and all rely on the user to have an existing installation of SuiteSparse. Furthermore, each of the three packages implements its wrappers using a different approach, making it infeasible for us to revive the existing projects or for a common community to maintain and extend them. We will create a unified SuiteSparse wrapper, distribute it with SuiteSparse itself, and provide a build system that ensures a seamless experience across all common operating systems and system architectures. This complete solution will be hosted in a single repository under the SciPy organization on GitHub. Mr. McKibben is an expert in wrapping such C++ code with Cython, as demonstrated in writing the wrappers underlying the SciPy interfaces in [22, 23, 21].

* We will benchmark the performance of these new features using Airspeed Velocity [27] to demonstrate that SciPy is a competitive, but free and open source, alternative to other popular numerical computing libraries.

We will spend 50% of our time on general maintenance, taming SciPy's growing backlog of pull requests in need of review and issues to be addressed. To complement the work proposed in our response to the CZI EOSS DEI program call, we will focus primarily on more technical contributions. For instance, PR 9523 [28] was submitted on November 22, 2018; its 1,612 new lines contain important corrections and improvements to SciPy's implementation of the Levy stable distribution, yet it has not received a complete review due to its large size and the sophisticated mathematics involved. Dr. Haberland has experience in pushing challenging PRs like this forward: for example, in the course of EOSS-0000000432, Dr. Haberland completed several stalled PRs [18, 19, 17], the oldest of which had been waiting for over six years. Funding this project would allow Dr. Haberland, his student research assistants, and Mr. McKibben to dedicate the time needed to perform the background research, thorough code review, and finishing touches such PRs deserve. In addition to seeking complex, high-impact PRs, we will also turn our attention to some of SciPy's oldest issues. Starting at the back (with an issue originally opened in 2006), we will solve problems that seem unlikely to be fixed otherwise. Resolving these old issues frees volunteer time for projects they find more exciting, improves project morale and health, and fixes bugs that have plagued SciPy users – and users of downstream libraries – for years.

The applicants will provide all required resources (e.g. computers, reference materials).

References

[1] X. S. Li, "An overview of SuperLU: Algorithms, implementation, and user interface," *ACM Transactions on Mathematical Software,* vol. 31, no. 3, pp. 302-325, 2005.

[2] T. A. Davis, "SuiteSparse: a suite of sparse matrix software," [Online]. Available: https://people.engr.tamu.edu/davis/suitesparse.html. [Accessed 17 May 2021].

[3] T. A. Davis, "Algorithm 832: UMFPACK V4. 3---an unsymmetric-pattern multifrontal method," *ACM Transactions on Mathematical Software ,* vol. 30, no. 2, pp. 196-199, 2004.

[4] T. S. Community, "Detailed SciPy Roadmap," [Online]. Available: http://scipy.github.io/devdocs/dev/roadmap-detailed.html. [Accessed 17 May 2021].

[5] Y. Chen, T. A. David and W. W. Hager, "Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate.," *ACM Transactions on Mathematical Software,* vol. 35, no. 3, pp. 1-14, 2008.

[6] T. A. Davis, Direct methods for sparse linear systems, Philadelphia: Society for Industrial and Applied Mathematics, 2006.

[7] M. Haberland, "ENH: added "interior-point" method for scipy.optimize.linprog," 12 August 2017. [Online]. Available: https://github.com/scipy/scipy/pull/7123. [Accessed 17 May 2021].

[8] M. Haberland, "ENH: optimize: added "revised simplex" for scipy.optimize.linprog," 12 January 2019. [Online]. Available: https://github.com/scipy/scipy/pull/9263. [Accessed 17 May 2021].

[9] A. Saad-Elind and M. Haberland, "ENH: Adds quadratic_assignment with two methods," 12 September 2020. [Online]. Available: https://github.com/scipy/scipy/pull/12775. [Accessed 17 May 2021].

[10] M. Haberland, "ENH: stats: add method of moments to rv_continuous.fit," 26 January 2021. [Online]. Available: https://github.com/scipy/scipy/pull/11695. [Accessed 17 May 2021].

[11] M. Haberland, "ENH: stats: add Page's L test," 27 January 2021. [Online]. Available: https://github.com/scipy/scipy/pull/12531. [Accessed 17 May 2021].

[12] M. Haberland, "ENH: stats: add Somers' D test," 29 January 2021. [Online]. Available: https://github.com/scipy/scipy/pull/12653. [Accessed 17 May 2021].

[13] M. Haberland, "ENH: stats: add Zipfian (different from Zipf/zeta) distribution," 31 January 2021. [Online]. Available: https://github.com/scipy/scipy/pull/13204. [Accessed 17 May 2021].

[14] M. Haberland, "ENH: stats: add method parameter to differential_entropy + more accurate estimators for small samples," 21 April 2021. [Online]. Available: https://github.com/scipy/scipy/pull/13845. [Accessed 17 May 2017].

[15] M. Haberland, "ENH: stats: add fast numerical inversion of distribution CDF," [Online]. Available: https://github.com/scipy/scipy/pull/13319. [Accessed 17 May 2021].

[16] M. Haberland, "ENH: stats: add bootstrap for estimating confidence interval and standard error of an n-sample statistic," [Online]. Available: https://github.com/scipy/scipy/pull/13371. [Accessed 17 May 2021].

[17] M. Haberland, "ENH: stats: add skewed Cauchy distribution," 9 January 2021. [Online]. Available: https://github.com/scipy/scipy/pull/13374. [Accessed 17 May 2021].

[18] M. Haberland and A. H. Wagner, "Update the Mann-Whitney-Wilcoxon test," 11 May 2021. [Online]. Available: https://github.com/scipy/scipy/pull/4933. [Accessed 17 May 2021].

[19] M. Haberland and J. Morton, "Permutation Ttest (new PR)," 11 January 2021. [Online]. Available: https://github.com/scipy/scipy/pull/4824. [Accessed 17 May 2021].

[20] L. V. Foster and T. A. Davis, "Reliable Calculation of Numerical Rank, Null Space Bases, Basic Solutions and Pseudoinverse Solutions using SuiteSparseQR," in *Householder Symposium XVIII on Numerical Linear Algebra*, Tahoe, 2011.

[21] N. McKibben and M. Haberland, "ENH: optimize: add HiGHS methods to linprog - continued," 15 November 2021. [Online]. Available: https://github.com/scipy/scipy/pull/12043. [Accessed 17 May 2021].

[22] M. Haberland and N. McKibben, "ENH: stats: add noncentral hypergeometric distributions (Fisher's and Wallenius')," 21 February 2021. [Online]. Available: https://github.com/scipy/scipy/pull/13330. [Accessed 17 May 2021].

[23] N. McKibben and M. Haberland, "ENH: Boost stats distributions," 8 May 2021. [Online]. Available: https://github.com/scipy/scipy/pull/13328. [Accessed 17 May 2021].

[24] The scikit-umfpack developers, "scikit-umfpack: The umfpack scikit provides wrapper of UMFPACK sparse direct solver to SciPy," 12 October 2018. [Online]. Available: https://github.com/scikit-umfpack/scikit-umfpack. [Accessed 17 May 2021].

[25] The scikit-sparse developers, "scikit-sparse: Sparse matrix tools extending scipy.sparse, but with incompatible licenses," 24 April 2021. [Online]. Available: https://github.com/scikit-sparse/scikit-sparse. [Accessed 17 May 2021].

[26] J. Jeronen, Y. Gingold and J. Bouas, "yig/PySPQR: Python wrapper for the sparse QR decomposition in SuiteSparseQR," 10 August 2020. [Online]. Available: https://github.com/yig/PySPQR. [Accessed 17 May 2021].

[27] The airspeed velocity developers, "Airspeed Velocity: A simple Python benchmarking tool with web-based reporting," 25 March 2021. [Online]. Available: https://github.com/airspeed-velocity/asv. [Accessed 2021 17 2021].

[28] B. Azzopardi, "ENH: improvements to the Stable distribution," 22 November 2018. [Online]. Available: https://github.com/scipy/scipy/pull/9523. [Accessed 17 May 2021].

8. *List expected milestones and deliverables, and their expected timeline. Be specific and include (where possible) any goals for metrics the software project(s) are expected to reach upon completion of the grant (maximum of 500 words)*

Dates assume a project start of September 1, 2021. PRs will be developed during the months indicated; a one-month opportunity for community review will begin in the following month. Dr. Haberland and Mr. McKibben are both SciPy core developers with commit rights; they will review the work of one another and the student assistants, and they will merge PRs after all feedback from the community review period is resolved.

September 2021 – December 2022: Write sparse Cholesky decomposition and symmetric linear system solver for distribution with SciPy. Compose symmetric sparse linear system benchmarks, and compare performance of new SciPy Cholesky features against equivalent Matlab operations.

January 2022 – August 2022:
Haberland*:* Write sparse QR decomposition and least-squares solver for distribution with SciPy. Compose sparse over- and under-determined linear systems benchmarks, and compare performance of new SciPy QR features against equivalent Matlab operations.
McKibben: Create a unified SuiteSparse wrapper and build system.

September 2022 – December 2023: Improve SciPy interface to UMFPACK. Compose asymmetric sparse linear system benchmarks, and compare performance of SciPy's UMFPACK interface to existing scipy.sparse.linalge.spsolve and equivalent Matlab operations.

January 2023 – May 2023: Add CHOLMOD interface to SciPy. Add SciPy's CHOLMOD interface to the symmetric sparse linear system benchmarks.

June 2023 – July 2023: Add SPQR interface to SciPy. Add SciPy's SPQR interface to the over- and under-determined linear systems benchmarks.

After completion of the proposed work, the SciPy library would have built-in capabilities for solving these fundamental sparse linear algebra problems and interfaces to more performant GPL-licensed solvers, all easily accessible on Windows, Mac, and Linux.

Throughout the year, the applicants would also devote 50% of their project time to SciPy maintenance, including responding to issues and reviewing PRs. Our goal is to close 200 issues and PRs by the end of the grant period. That is, although 1200-1600 Issues and PRs are closed (or merged) annually; we will be significantly involved in the resolution of at least 200 of these (in addition to those related to the work above).

9. List active and recent (previous two calendar years) financial or in-kind support for the software project(s), including duration, amount in USD, and source of funding. Include in this section any previous funding for these software projects received from CZI (maximum of 250 words)

SciPy was awarded $200k in CZI EOSS Cycle 1 for the project "A Solid Foundation for Statistics in Python with SciPy" (EOSS-0000000432). The original duration of the grant was 1 year, and a four-month no-cost extension was approved.

SciPy has received several short-term Small Development Grants from NumFOCUS:

Add PROPACK Sparse SVD to SciPy – $3550 – 2021
Improving Boundary Handling and Data Type Support in scipy.ndimage – $5000 – 2000
Enhanced LAPACK Support in SciPy – $4978 – 2019
Complete the SciPy Special Functions Documentation – $2500 – 2019
SciPy Development Documentation Overhaul – $4274 – 2019
An Efficient, High-Level Implementation of Linear Programming – $2000 – 2018
Maturing a Sparse Array Implementation for SciPy – $3000 – 2018

SciPy has received $2500 per month from Tidelift since June 2019. This funding is slated for work not proposed here, such as a redesign of the SciPy website.

10. *Landscape Analysis: Briefly describe the other software tools (either proprietary or open source) that the audience for this proposal primarily uses. How do the software projects in this proposal compare to these other tools in terms of user base size, usage, and maturity? How do existing tools and the project(s) in this proposal interact? (maximum of 250 words)*

IDL and Mathematica are alternatives for many scientific tasks, but neither have the sparse linear algebra support that we propose. With all its toolboxes, Matlab exceeds the capabilities of SciPy in many areas, including the sparse linear algebra features we propose to add. However, none of these options are free or open source, which can be major barriers to their use. Regarding interaction, it is common for users of these languages to transition to Python/SciPy when they need free software to perform similar tasks. To ease the transition, we consider their APIs (especially Matlab) when designing the interface to new functions. The R language is open source and has many of the proposed features available in its "Matrix" library, but Python is a much more popular language outside of statistical applications.

The SuiteSparse C++ library has bindings for use in several languages, including Python, but these bindings are not well documented and maintained. Other sparse linear algebra libraries (e.g. Eigen) exist, and some have more mature Python wrappers, but SuiteSparse is generally regarded as the most performant library. For these reasons, it is important to add 1) custom, permissively licensed sparse linear algebra solvers and 2) wrappers for SuiteSparse routines to SciPy, the only popular foundational mathematical algorithms library for Python, with a twenty-year history, ten thousand dependent packages, hundreds of thousands of dependent repositories, and millions of users.

11. Value to Biomedical Users: Briefly described the expected value the proposed scope of work will deliver to the biomedical research community (maximum of 250 words)

Sparse linear algebra is used directly in some biomedical research including analysis of conservation laws in large biochemical networks [1], and sparse linear algebra lies at the heart of the solution of partial differential equations, which have many applications in biomedical engineering [2]. Improvement of SciPy's sparse linear algebra capabilities expands the capabilities of other CZI-funded projects that rely on them, such as scikit-image. Moreover, as SciPy is a fundamental software library with thousands of dependent projects, the general SciPy maintenance work supported by this grant will benefit the biomedical research community through its dependent libraries.

[1] Vallabhajosyula, Ravishankar Rao, Vijay Chickarmane, and Herbert M. Sauro. "Conservation analysis of large biochemical networks." Bioinformatics 22.3 (2006): 346-353.

[2] Schiesser, William E. Partial differential equation analysis in biomedical engineering: case studies with MATLAB. Cambridge University Press, 2012.

12. *Advancing DEI is a core value for CZI, and we are requesting information on your efforts in this area. Describe any efforts the software project(s) named in this proposal have undertaken to increase diversity, equity, and inclusion with respect to their contributors and audience. Please see examples from applications funded in previous cycles (maximum of 250 words)*

As an open-source project, anyone and everyone is welcome to contribute to SciPy. Accordingly, the SciPy project adopted the following diversity statement shortly before the release of SciPy 1.0 in late 2017:

"The SciPy project welcomes and encourages participation by everyone. We are committed to being a community that everyone enjoys being part of. Although we may not always be able to accommodate each individuals preferences, we try our best to treat everyone kindly.

No matter how you identify yourself or how others perceive you: we welcome you. Though no list can hope to be comprehensive, we explicitly honour diversity in: age, culture, ethnicity, genotype, gender identity or expression, language, national origin, neurotype, phenotype, political beliefs, profession, race, religion, sexual orientation, socioeconomic status, subculture and technical ability, to the extent that these do not conflict with this code of conduct."

Nevertheless, the group of SciPy contributors is not as diverse as it should be. As another step to increase the diversity of the SciPy community, we hosted a mentored SciPy development sprint for diverse beginners at PyCon 2021 as part of the CZI EOSS Cycle I project. We aspire toward more substantial change: we are currently submitting a separate proposal for the CZI EOSS DEI program with NumPy, matplotlib, and pandas which would create two cross-project positions dedicated to contributor experience. Through these positions, we hope to make lasting, positive impact on DEI in open source scientific software.

9. Open Source Project #1 Details:

1. Project Name: SciPy (Library)

2. Homepage URL: https://www.scipy.org/

3. Hosting Platform: GitHub

4. Main Repo: https://github.com/scipy/scipy

5. Short Description of Project:

SciPy is a library of numerical routines for the Python programming language that provides fundamental building blocks for modeling and solving scientific problems. SciPy includes algorithms for optimization, statistical analysis, integration, interpolation, eigenvalue problems, differential equations, fast Fourier transforms and many other classes of problems; it also provides specialized data structures, such as sparse matrices and k-dimensional trees. SciPy is built on top of NumPy, which provides array data structures and related fast numerical routines, and SciPy is itself the foundation on which higher level scientific libraries, including scikit-learn and scikit-image, are built.