Project Summary

The MPS and CISE directorates are most relevant for this proposal.

Several billion dollars of NSF investment into research facilities and projects critically rely on the scientific Python ecosystem (SPE) for data analysis and scientific computing, including large telescopes (DKIST, LSST, TMT), the Advanced Laser Interferometer Gravitational Wave Observatory (AdvLIGO), the Ocean Observatory Initiative and many others. However, there has been no direct government funding for the core projects of the SPE (numpy, scipy, matplotlib and pandas), which are central infrastructure for large parts of NSF's research portfolio. This proposal brings together core software developers and project leads from the five most commonly used packages for scientific computation in Python; we propose to fund research software developers on these projects to ensure the continued improvement of these computational tools, and to ensure the ecosystem evolves to meet the growing needs of the scientific community served by the NSF. The NSF is in a unique position to fund these fundamental computational tools that have a broad impact across all sciences and engineering disciplines. If this opportunity is missed, the ecosystem will need to rely on industry funding for further development, and project road maps will be aligned with the needs of the software industry, not the sciences.

While this ecosystem contains hundreds of specialized packages, there is a core of central infrastructure packages on which most other tools. These are **NumPy** for numeric data structures (nd-arrays); **SciPy** for a collection of scientific needs like linear algebra, statistical methods, signal processing and optimization, **Matplotlib** for visualizationl **pandas** for handling tabular and heterogeneous data, and **scikit-learn** for machine learning. This proposal brings together the core contributors to this essential infrastructure to further integrate these components and enhance them to cater to the evolving needs of scientists. Key enhancements that we are proposing are tighter ingegration of the libraries, **performance improvements, in particular multi-core support**, infrastructure to **support measurement units**, a more consistent **data model for visualization**, better support for **heterogeneous hardware** and **unified documentation**, to foster even broader and more effective use by the scientific community.

Intellectual Merit

This proposal creates a tie between the scientific community and the SPE by involving key stakeholders in both. This will ensure that future development of the SPE will match the dynamic needs of the scientific community supported by NSF. The proposal will also accelerate the implementation of key features of the SPE that will directly benefit scientists across many disciplines. The proposal also brings together the core projects of the SPE, which will allow more coherent planning and implementation, improving usability and reducing complexity. Together, these contributions will expedite scientific discovery in a wide range of fields.

Broader Impacts

By providing high quality tools free of license fees and onerous usage terms, the SPE has enabled a generation of domain experts, e.g. journalists, crime analysts, etc, to use computational thinking in their work. By building this critical infrastructure for science, we democratize the access to computational thinking and provide society with a greater ability to analyze events. This proposal will further promote the role of the research software engineer in the university that has recently become identified as a core need by the academic community. The proposal also includes measures to directly improve diversity in participation in scientific open-source projects through a student internship program and a comprehensive assessment program and to improve the accessibility of scientific visualizations to people suffering from colorblindness.

Project Description

This project is a proposal for the Mid-scale RI-1 "Implementation" award.

1 Introduction

> Python has arguably become the *de facto* standard for exploratory, interactive, and computation-driven scientific research. Millman and Aivazis [26]

Several billion dollars of NSF investment into research facilities and projects critically rely on the scientific Python ecosystem (SPE) for data analysis and scientific computing, including large telescopes (DKIST, LSST, TMT), the Advanced Laser Interferometer Gravitational Wave Observatory (AdvLIGO), the Ocean Observatory Initiative and many others. However, there has been no direct government funding for the core projects of the SPE, which are central infrastructure for large parts of NSF's research portfolio. This proposal brings together core software developers and project leads from the five most commonly used packages for scientific computation in Python; we propose to fund research software developers on these projects to ensure the continued improvement of these computational tools, and to ensure the ecosystem evolves to meet the growing needs of the scientific community served by the NSF. The NSF is in a unique position to fund these fundamental computational tools that have a broad impact across all sciences and engineering disciplines. If this opportunity is missed, the ecosystem will need to rely on industry funding for further development, and project road maps will be aligned with the needs of the software industry, not the sciences.

While scientific computation is performed on many different platforms and using a broad range of cyber infrastructure, the Python language and its associated packages and resources have become increasingly popular across several sciences [26]. The tools available for scientific computation and data analysis in Python are one of the key reasons scientists of many disciplines choose this framework, collectively known as the Scientific Python Ecosystem (SPE; also PyData when emphasizing the data analysis aspects, often referred to just as "Python"). This ecosystem is used by large scientific projects from Astronomy and Physics to Biology and Geosciences and makes up essential cyberinfrastructure for these projects.

The Scientific Python Ecosystem is a loosely defined community of projects and programmers with the common goal of advancing the use of the Python programming language in Science. This development is largely volunteer work, or work that is sponsored implicitly by specific science projects. While this ecosystem contains hundreds of specialized packages, there is a core of foundational packages on which most other tools rely. These are **NumPy** for numeric data structures (nd-arrays), **SciPy** for scientific algorithms like linear algebra, statistical methods, signal processing and optimization, **Matplotlib** for visualization, **Pandas** for handling tabular and heterogeneous data, and **scikit-learn** for machine learning. This proposal brings together the core contributors to this essential infrastructure to further integrate these components and enhance them in order to better serve the evolving needs of NSF-supported scientists. Key enhancements that we are proposing are tighter integration of the libraries, **performance improvements, in particular multi-core support**, infrastructure to **support measurement units**, a more consistent **data model for visualization**, better support for **heterogeneous hardware** and **unified documentation**, to foster even broader and more effective use by the scientific community.

2 Current Usage and Impact

> The scientific Python ecosystem is critical infrastructure for the research done at LIGO.
> David Shoemaker, LIGO Scientific Collaboration

While large parts of the scientific computing community rely on the SPE, the open nature of these infrastructure tools makes measuring their influence hard. We expect that a large fraction of NSF-sponsored projects rely on this shared infrastructure, but this is hard to prove in practice. In particular, scientific work does not usually cite the software that was used for computation, and usage and download statistics are hard to gather for widely used open source packages. Even though citation counts are likely to vastly under-represent the scientific use of these packages, we want to point out that the reference for scikit-learn [30] has over 14,000 citations, Matplotlib [16] over 6000 citations, SciPy [18] over 4000 citations, Pandas [25] over 1000 citations and a common reference of NumPy[39] has over 3200 citations. These counts already illustrate a problem in measuring usage using citations, as every user of scikit-learn also uses NumPy, but scikit-learn has over four times as many citations. Similarly, citations for domain-specific tools like Qiime [6] for gene sequencing (13,500 citations), PHENIX [2] for X-ray crystallography (14,000 citations) and AstroPy [36] for astronomy (about 2000 citations) often have high citation counts compared to the core packages like NumPy, SciPy and Matplotlib that they are built upon. The Astronomy community however is quite explicit in their reliance on the SPE and the Thirty Meter Telescope (TMT, $1B in NSF funding), the James Webb Space Telescope (JWST) and the Large Synoptic Survey Telescope (LSST, $470M in NSF funding), which all selected Python and SPE as the standards for their data analysis pipelines [40]. Similarly LIGO relies critically on the SPE for data analysis (see quote above, over $1B in NSF funding). As a matter of fact, the experiment proving the existence of gravitational waves was made its data public together with data analysis Python code using NumPy, SciPy and Matplotlib less than a day after announcing their finding. The National Energy Research Scientific Computing Center (NERSC) found that about half of the users of their supercomputer use Python, and that NumPy, SciPy, Matplotlib, Pandas and scikit-learn are the five most frequently used Python libraries among NERSC users [7]. In neuroscience, SPE is emerging as the basis of a shared ecosystem [31]. In bioinformatics, the Galaxy software [3] has been used in over 7000 papers and makes extensive use of SPE, as does the $50M IPlant collaboration [17]. In geoscience, scientists at the Ocean Observatories Initiative ($330M in NSF funding) rely heavily on the SPE for data analysis [10]. The NSF's EarthCube program, which supports cyber-infrastructure for geosciences, also leverages the SPE heavily in projects such as Pangeo, netCDF, Ensemble Toolkit, and Drilsdown. The Journal of Open Source Software found that of the 111 papers published in the first year, 54 were describing new research software based on Python and the SPE [33], and an analysis of the arXiv pre-print server showed that at least 13% of papers posted to the platform contain a plot generated with Matplotlib. Given this wide-spread use of SPE in many areas of science, it should be clear that SPE is critical cyberinfrastructure in diverse fields from Genetics to Astrophysics, and that research projects and facilities relying on the SPE represent several billion dollars of NSF funding.

3 Proposed Activities

The majority of our effort will be spent on technical improvements of our projects and tighter integration between them - **building computational infrastructure**. In addition we will spend significant effort on improving educational materials, as well as specific initiatives to broaden and diversify our community and better connect academia to the scientific open source community - **building community as infrastructure** [4].

3.1 Technical improvements

Performance improvements Speed improvements, lower memory usage and the ability to parallelize algorithms are beneficial to most science domains and use cases. Therefore performance will be one of the main themes of the technical work we propose; we'll focus on this for the

complete duration of the program. We identify a number of concrete work items here which will be augmented by user feedback:

Parallelization interface in Pandas, SciPy and scikit-learn. The original SPE code did not emphasize parallel execution. However, over the past decade, trends in microprocessor architecture indicate that clock speeds are stagnant but the numbers of CPU cores per chip are increasing [32]. The core SPE routines therefore need to be updated to take advantage of modern hardware. We will extend the number of algorithms that are able to run on multiple CPU cores (currently very few in SciPy and Pandas). We will design a new parallelization interface for this that is consistent between the projects, as well as compatible with higher level parallel libraries (Dask, Xarray).

Performance benchmarking suites will be added to each project. Those will be triggered on a weekly basis and the results will be published automatically to a website, with any performance regressions flagged for investigation. Improving NumPy performance will have a positive impact on almost every use case. We will optimize the performance of key NumPy array operations (e.g. indexing) and internals (e.g. strategic use of newer CPU hardware acceleration features).

Pandas data store performance . Pandas was designed internally for optimal performance (zero copying) on 2D NumPy arrays. However, Pandas dataframe construction from 1D arrays has become a widespread use-case, which we will optimize.

Accelerate SciPy algorithms with Numba. Numba is a just-in-time compiler for Python code, which can give very large speedups. Currently Numba is not yet used within SciPy or any of the other core SPE projects, however it has rapidly become popular in end user code. We will introduce a Numba dependency in SciPy, and accelerate key algorithms for numerical optimization, integration and signal processing.

Arrays with physical units. NumPy arrays typically have numerical (e.g. 64-bit float, 128-bit complex), string or date-time (e.g. nanosecond) elements. NumPy is in the process of adding a new mechanism to allow users to easily add custom data types. This opens up many new possibilities, in particular storing arrays with physical units. There are currently many competing implementations of physical units calculation frameworks, including astropy.units, pint, yt and quantities. However due to the lack of NumPy data types these are cumbersome to use and not well integrated. The new custom data type support in NumPy allows us to create a better implementation, which we we will then integrate with core packages like Matplotlib and Pandas.

A sparse n-dimensional array data structure. SciPy provides sparse matrices, which are a fundamental data structure for many applications, from computer graphics to genomics to partial differential equations and are heavily used for machine learning. SciPy currently provides only a 2-D sparse array, with an interface that's inconsistent with NumPy arrays. We will implement a new sparse n-D array, fully consistent with NumPy, in SciPy and integrated with scikit-learn.

Distributed and GPU array support. Hardware for scientific computing is becoming more heterogeneous every year. Users are often requesting better support for their particular hardware, e.g. for HPC systems [37] or deep learning applications [20]. We will meet this need by integrating NumPy with distributed and GPU-based array execution engines (Dask and CuPy and PyTorch respectively).

Higher-level plotting abstractions. Matplotlib is an incredibly flexible tool for creating 2D visualizations with a high level of control by the user. However, this flexibility comes with a high degree of complexity that can make creating complex interactive visualizations cumbersome. We will improve this via:

Unifying the internal data abstractions We will create uniform high-level abstractions for representing data and visual styles. These will allow more generic interfaces in

dealing with data, which enable embedding data into figures and collaborative creation of visualizations. It will also ease working with data streams and provide a library agnostic interface to interact with other libraries like Bokeh, Plotly and Altair.

Defining formal conventions for Matplotlib extensions, which will simplify and unify the interface of existing, domain-specific visualization tools built on top of matplotlib and enable cross-domain and multi-modal science.

Easing color-blind accessible visualization. The SPE has long made it a priority to provide tools for people affected by colorblindness [34]. However, designing visualizations in a way that is accessible for those affected by colorblindness still requires some effort by the scientists creating a visualization. We will implement color-blind accessible defaults across the SPE, so that most figures will be color-blind accessible out-of-the-box. We will also provide tools to quickly assess if a given figure is colorblind-friendly or not by emulating common conditions.

Cython usability improvements. Cython is a language with a Python-like syntax that compiles to C code that can then be used from Python. Due to its speed, it is heavily used in SPE, as well as in many domain-specific scientific packages. Scientific users often find it hard to extract the performance gains from Cython that are possible. We will address this via key usability improvements to Cython when used in combination with NumPy and other SPE packages - in particular improving diagnostics and documentation.

Cross-stack integrated testing. Each project has its own continuous integration (testing) system, which aims to ensure that each code and documentation change does not break anything. In practice however this is not enough. End user code relies on a chain of dependencies and changes in, say, NumPy can propagate through SciPy, Matplotlib or Pandas to domain-specific libraries and the end users' code. We will build a system that tests both the latest changes and stable releases of all of the SPE projects together as well as lower-level dependencies (Cython, OpenBLAS). The system will alert projects of potential issues and allows them to fix those before they disrupt the workflow of scientists.

3.2 Educational materials

Well-written and complete online documentation is essential for users. It is by far the most important form of information for users to learn how to effectively use their computational tools. For example, the SciPy documentation website alone gets 6.5 million visits per month - this is several orders of magnitude more exposure than the most popular books, webinars and online courses. Online documentation is also the best format to enable continuous updating by the community, so the content stays relevant. Each project has comprehensive reference documentation. However, reference documentation is written to show the capabilities of the packages, not to address a specific user need, which usually requires several packages. We will employ a technical writer, a role typically not present in volunteer-only projects, to work with domain experts for each project and create a complete set of tutorials and guides, in a uniform style. We will focus on *cross-project documentation* that helps scientists get started, teaches them best practices and how to navigate the large ecosystem of scientific projects. This cross-project documentation will function as a central point of entry to the ecosystem, which does not exist today, and address several major user groups: beginning scientists, educators, users who want to become contributors, and domain experts (e.g. in visualization, high-performance computing, or life sciences). Beyond content creation, we will review existing documentation to identify needed accessibility and quality improvements.

3.3 Community Building

While the SPE has matured to serve scientists in all disciplines organically, development should be catered to the diverse scientific community. To that end we propose several types of outreach including a stakeholder advisory group, a needs analysis overseen by major scientific projects, an

annual public user survey for feature needs and improvements, and inclusivity surveys to understand the gaps in diversity of users and contributors to the SPE. To strengthen direct collaboration with science end-users, this proposal includes Prof. Szabolcs Márka of Columbia, working on multimessenger Astrophysics, and Prof. Ryan Abernathy of Lamont-Doherty Earth Observatory, a computational oceanographer, as co-PIs. These are prime examples of scientific research groups being empowered by the use of the SPE. Partnering with scientific groups that are tightly integrated into their respective communities (astrophysics and earth and environmental sciences respectively) will allow us to get direct feedback about implemented improvements and requested features on an ongoing basis.

Our efforts in outreach and inclusivity will be led by a program manager and a community assessment coordinator. We will also consult with a sociologist in order to ensure the surveys are conducted in a rigorous and responsible fashion and that the results are interpreted properly.

Stakeholder advisory group The outreach for this grant produced numerous scientists interested in investing time supporting and improving the SPE. To continue to focus our work on NSF science priorities, we propose convening a committee of scientific advisors from key partner projects including, but not limited to, LIGO, IPlant, LSST, OOI, EarthCube, NCAR and similar groups (full stakeholder list will be confirmed for the final proposal). This committee will be charged to review major milestones and produce plans for prioritizing work. This will include a needs assessment in the represented disciplines and ensuring that these needs are met by the SPE project roadmaps. An annual one day workshop will be held at an SPE conference such as the annual SciPy meeting, to maximize the interaction with SPE core developers.

Public Survey The user community of the SPE goes much beyond the large NSF programs; directly engaging with every research group relying on the SPE is not feasible. Therefore we propose hosting a large, well publicized survey for users and beneficiaries of the SPE. This annual survey will serve as the launching point for understanding the unmet needs of the community and will inform the work of the SPE for many years to come.

Inclusivity Assessment There is strong evidence that diverse contributors and community members produce more innovation [11]. However, there is an acute absence of existing survey information on the demographic makeup of discrete open source project communities, as well as a lack of information about demographics in open source participation. Without better information about the demographics and inclusive practices (or lack thereof) in specific open source communities, it is difficult to assess the impact of diversity and inclusion practices and programs, much less to identify targeted best practices. The GitHub Open Source Survey for 2017 [41] is frequently cited as one of the only recent survey efforts aimed specifically at the open source community, yet that survey only has 5,500 respondents on a platform with millions of users.

We propose development and implementation of a survey instrument to assess diversity demographics and inclusive practices among all 48 NumFOCUS projects. Results of the survey will be used to identify the more successful diversity and inclusion measures and practices each project has used and encourage their implementation in other projects. The target community for this survey includes thousands of open source contributors across the globe. We plan to use results from the survey to develop educational materials and training curricula to help educate open source project communities on best practices for inclusion and sustaining participation from diverse individuals. We will be able to measure the impact of these interventions on NumFOCUS sponsored projects with an eye towards iterative improvement of the educational materials and refining best practice. The benefits of the survey results will not be limited to NumFOCUS projects, as we will make the insights gleaned from the survey and the educational materials publicly available.

Student Internships for Open Source Few educational experiences can compare to the act of

doing. To this end, we propose 5 undergraduate students a year with a focus on finding 40-60% from underrepresented groups. Similar to the focus of an REU for research, these students will actively participate in creation of research infrastructure, under the guidance of developers and scientists alike. Lessons won from our community's 15 years of participating in programs like Google Summer of Code lead us to believe there is a gap in funding students for the dual purpose of software development and research infrastructure. We propose a new program modeled after the Jupyter project at CalPoly, with an ongoing internship that focuses on students implementing necessary features with a goal of merging into a main line projects.

This program will be led by Dr. Andreas Müller and Professor Szabolcs Márka jointly at Columbia, where there is institutional support for student positions. This activity will not only help build critical infrastructure, but also set a proving ground for the documentation efforts. For what better way to ensure completeness and readability than requiring a set of novices to use it for implementation. This internship will also provide the students with dual mentorship in software development and research. This program will help fill the educational gap and equip students with the practical scientific computational skills they need.

The students will each be required to submit a proposal based on specific ideas from the project road map. A member of the grant staff will be assigned to each student to guide them through the process of implementing and submitting the code to the community. Dr. Müller and Professor Márka will oversee the program as a whole, ensuring mentorship for each student is met each according to the appropriate need.

3.4 Milestones and Timeline

Table 1 shows an overview of major milestones, which will be reviewed at an annual workshop.

4 Intellectual Merit

This proposal creates a direct tie between the scientific community and the SPE by involving key stakeholders in both. This will ensure that future development of the SPE will match the dynamic needs of the scientific community supported by the NSF. The proposal will also accelerate the implementation of key features of the SPE, as outlined above. Without the proposed work, addition of these features would rely on volunteer work and might not be implemented for many years, or at all. The proposal also brings the core projects of the SPE together under a common umbrella, which will allow more coherent planning and implementation. This will increase usability and reduce complexity of the SPE, which in turn will improve the workflows of all scientists relying on the SPE. This will facilitate quicker turn-around on data analysis as well as shorter on-boarding procedures for researchers new to a particular project or software infrastructure.

4.1 Scientific Justification

This proposal will accelerate scientific discovery in a wide range of fields by improving computational tools, as well as instructional material for learning them. Lack of computational skills has been identified as the main bottleneck in applying data-driven methods in biology [35], environmental science [23] and others. We addresses this in three ways:

- Improving coherence and ease of use of the SPE libraries for common scientific task.
- Creating additional instruction material directly targeted at scientists.
- Embedding experts in computational tools in research institutes to engage with scientists.
- Providing students with software-development experience via summer internships.

Improved open source tools and documentation thereof will also improve reproducibility of scientific results made using these tools [35, 23, 24, 4, 12, 19].

Year	Milestone
Year 1	Parallelization interface for SciPy, Pandas Performance benchmarking suites (all SPE projects) First NumPy performance optimizations shipped to end users Results from first user survey published (repeats every year) Inclusivity survey results published (repeats in years 3 and 5)
Year 2	Sparse n-dimensional array data structure implemented SciPy optimization algorithms accelerated with Numba Pandas data store performance optimizations Conventions for Matplotlib extensions Consistent multi-core support in scikit-learn Cross-project documentation published (repeats in later years)
Year 3	Numba acceleration of SciPy signal processing and numerical integration algorithms Color-blind accessible defaults released Cython usability improvements Cross-project integrated testing system
Year 4	Arrays with physical units User-editable Matplotlib figure objects Consistent multi-core support in Pandas and SciPy Impact survey results and methodology published
Year 5	Sparse n-dimensional array data structure integrated in SciPy and scikit-learn Streaming plots & animations GPU & distributed array support in NumPy Project close-out documents

Table 1: Key Project Milestones. Incremental deliveries happen earlier.

4.2 Research Community Priority

NSF's "Vision and Strategy for Software for Science, Engineering, and Education", states that software is a critical and pervasive component of cyberinfrastructure for science, engineering, and education. As argued in above, the SPE in particular is critical infrastructure for many sciences, both for research and education. We expect that the usage of SPE in the sciences will be increasing in the coming years, given that some of the projects mentioned above, such as the TMT, will become operational only in 2030. Several of the NSF Big Ideas are directly tied to use of the SPE and would benefit from the proposed improvements, tighter integration of packages and more direct communication between scientists and open source developers. The following Big Ideas are particularly relevant for this proposal:

Growing Convergence Research Given the ubiquity of Python and the SPE, the SPE already is the common language for data analysis across disciplines [26]. Further integration and unification of the SPE will result in even better interoperability and convergence on the level of software tools and methods.

Harnessing the Data Revolution A critical component for widespread use of data driven science is the availability of accessible software tools, and instructions that equip scientists with the necessary computational literacy. The SPE is already the basis of much data sciences education and data driven research [26, 27], and our proposed activities will further increase usability, provide documentation directly aimed at scientists, and increase computational competency.

Windows on the Universe As mentioned in "Current Usage and Impact", the SPE is the basis of the data analysis tools used in most of astrophysics, particle astrophysics (e.g. IceCube experiment), and is central to the cyber-infrastructure used for the discovery of gravitational waves at LIGO. Professor Márka will provide a direct link to this community and ensure that prioritization within the SPE matches the needs of the multimessenger astrophysics community.

NSF Includes Open source has been identified as a driver of scientific discovery [23], and it can be expected that inclusivity problems and practices in the open source community reflect those in science and engineering. The proposed inclusivity survey will provide a clear measure of the state of several projects, as well as create recommendations for inclusivity practices. The skills acquired by participating in open source are highly valuable for academic and engineering careers and improving diversity in projects will have a positive impact on diversity in these areas as well.

Our specific work items are derived from our interactions with the research community, and while we hope that this proposal will establish a closer connection between the communities, the work items reflect existing priorities, in particular better use of multi-core systems [38, 21, 37], integration of matplotlib with other plotting tools [22] and ability to collaborate on plots [13], broad support for sparse n-d arrays [22, 7] and better integrated support for physical units [9]. Improved benchmarking and integration testing, as described in the the technical improvements, are important to ensure the long-term stability of the SPE.

4.3 Establishing Research Software Engineers

It has been observed that in many areas of research, a lack of computational knowledge is hindering innovation [28, 5]; this has been particularly obvious in data driven research [29]. One proposed component of increasing computational literacy to meet the goals of effectively "harnessing the data revolution" that has been found effective [28, 19] is introducing new roles focused on software engineering and computational aspects of science. We will refer to these as Research Software Engineers, though the position is very related to the Data Scientist role described in [28]. These positions represent the core of this proposal, which establishes 8 new medium-term roles across several organizations. Embedding these Research Software Engineers will have network effects, improving computational literacy in these organizations and modelling a new career track that is currently lacking in many academic institutes in the US. The time-frame of this proposal will allow the participating organizations to observe and evaluate the influence of these new roles, and will hopefully lead to the creation of more permanent research software engineer positions.

5 Broader Impacts

By providing high quality tools free of license fees and onerous usage terms, the SPE has enabled a generation of domain experts to use computational thinking in their work. For example the LA Times, as well as many other journalism organizations, has been able to build "data desks" equipping their journalists with computation as a primary driver in their investigations [1], and the SPE has been instrumental in analysis of online criminal behaviour by the criminal justice community [8]. By building this critical infrastructure for science, we democratize the access to computational thinking and provide society with a greater ability to analyze events.

Our improvements to the accessibility of scientific visualizations for people suffering from color-blindness will remove every-day hurdles for scientists and students suffering from this condition. Past changes to the default styles in the SPE have found wide adoption [34], and the proposed changes will propagate to a wide array of scientific publications and instructional materials.

By building well-written and complete cross-project online documentation and tutorials, we

lower the barriers for the domain expert to use the SPE. The U.S. research community will be able to utilize, update, and maintain these documents for decades with much less effort than the initial creation. This will lessen the burden of maintenance on software critical to scientific discovery, increasing the ability of the community to reproduce prior work.

The proposed internship and survey activities are intended to increase diversity of underrepresented groups, document the current state of affairs, and create assessment data to improve future programs. By creating surveys that can be administered by professional staff both before, during, and after the program, we will create an authoritative dataset that can be used to answer questions about inclusion programs in open source software communities. While proxy datasets exist, they are well known to be problematic as they usually do not follow best practices for social science research. Additionally, internships are a great entry point for cultivating the next generation of leaders, as they are provided the resources to overcome many of the challenges to entering the field. By locating the internship at Columbia University, the interns will have unparalleled access to the leaders of science giving opportunity for lifelong mentorship.

6 Ongoing operations and maintenance plan

Our long-term goal is to grow both the contributor base and the organizational maturity of SPE projects to the level needed to continue sustaining innovation. The SPE grew into the computational tool set of choice for large areas of science over the last decade largely based on volunteer effort. NumPy received one $1.1M grant from the Sloan and Moore foundations [15] that allowed the project to hire 2 engineers until the end of 2019; scikit-learn received one $400k grant from the NSF; other projects are fully volunteer-driven. **Given the explosive growth of the user base, volunteer-only projects at the core of the ecosystem are not sustainable.** The funding we request in this proposal will be used to fund two developers per project. This will allow addressing critical performance bottlenecks and major features that likely would not be implemented with volunteer-only effort, or require a time span that is not in line with scientists' needs. Additionally, a limited amount of funded developers has the potential to unlock additional volunteer effort, as evidenced by the NumPy Sloan grant.

We aim to diversify funding sources over the next 5 years. We will focus primarily on major users from industry. Prominent industrial users, especially in high tech (Microsoft, IBM, NVIDIA, Netflix) and in finance (Bloomberg, Capital One, Two Sigma) have recently started sponsoring either a specific project or the wider ecosystem via NumFOCUS. Furthermore, SPE underpins the current wave of AI development - e.g. the flagship AI projects from Google (Tensorflow), Facebook (PyTorch) and Amazon (MXNet) all depend on it. We expect this AI ecosystem to continue growing at a rapid pace and serve as an increasingly important source of funding. Monetary funding from scientific users is harder; however, there is growing awareness that research software engineers are important. RSEs for projects that rely on SPE can and do contribute to the code of our projects, which constitutes indirect funding. **This diversified funding is not yet close to what we need, however over 5 years we are confident we can grow it to, at a minimum, a replacement for the funds requested in this proposal.** We do not foresee any request to the NSF outside of the Mid-scale RI program. Instead, we will focus on diversifying our funding sources, as well as on expanding our current volunteer contributor base.

Metrics that we will track over time to measure the impact of our proposed work are the rate of increase of unique contributors, and retention of existing contributors [14], diversified funding levels ($/year), rate of change of issues opened/closed and performance improvement (%) on benchmark suites. In addition we will compile the qualitative and quantitative feedback from the yearly user survey.

7 Project Team and Collaborations

This proposal is unique in that it brings together leading figures of all the core projects of the SPE, and has widespread support both in the Python open source community as well as in the scientific community. The **NumPy** and **SciPy** projects are represented by **Ralf Gommers**, one of the lead developers of NumPy and Chair of the SciPy steering council, who will lead the contributions to these projects and **Travis Oliphant**, the creator of NumPy and SciPy, who will be serving in a consulting role. The **Matplotlib** project is represented by the project leader (BDFL), **Thomas Caswell**, who has been working on the project for 7 years and in a leadership role for 4 years, and will lead contributions to Matplotlib and be involved in visualization contributions in general. The **Pandas** project is represented by **Jeff Reback**, who has been leading the project for 6 years. The **scikit-learn** project is represented by PI **Andreas Müller**, who has been a core-developer for over 8 years. This group of people is ideally positioned to make ecosystem-level contributions to the SPE and coordinate efforts across packages. Given the complexities inherent in major updates to each core SPE project, direct involvement of the project leaders is necessary for the success of any collaborative effort at this scale.

The proposal also includes **Szabolcs Márka**, providing a link to the astrophysics community and **Ryan Abernathey**, providing a link to the research community in Earth and Environmental Sciences. Abernathey is also a core-developer of the xarray Python package, a library within the SPE which is crucial for geoscience research and which relies heavily on the core packages discussed in this proposal.

Parts of this proposal will be executed by sub-awards to NumFOCUS and QuanSight Labs. NumFOCUS is a non-profit umbrella organization for open source projects related to scientific computing and sponsors 27 projects within and beyond the SPE, including Project Jupyter, the Julia language and the rOpenSci project. NumFOCUS has programs for education, diversity, sustatinability, and fiscal sponsorship for open source scientific software projects. Dr. Gina Helfrich, NumFOCUS Program Manager for Diversity and Inclusion, will serve as co-PI and community assessment coordinator. Dr. Helfrich was the former director of the Harvard College Women's Center and an experienced consultant on diversity and inclusion issues. QuanSight Labs was created as a home for SPE core developers by the creator of NumPy and SciPy (Travis Oliphant), and employs a significant fraction of founders and leads of SPE projects.

8 Related Projects and Products

Open source projects have been essential to computational science, replacing commercial tools for scientific computation like Matlab and Mathematica and commercial statistical software like STATA and SPSS [4, 12, 23, 26, 24]. While the SPE plays a central role in this development, there are other important open source projects in this space, most prominently the R ecosystem for statistical computing [4] and the Jupyter project that provides a browser-based literate coding and data exploration environment. We believe both of these to be similarly critical infrastructure, with R serving slightly different communities and needs than Python, focusing more on statistical methods and less on general computation, and Jupyter providing a user-friendly interface that supports interactive and collaborative research for Julia, Python, R (hence JuPyt[e]R) and other languages. However, this proposal focuses on the highly interdependent SPE, with cross-project objectives that are specific to the SPE.

References Cited

[1] (2019). Los angeles times data desk.

[2] Adams, P. D., Afonine, P. V., Bunkóczi, G., Chen, V. B., Davis, I. W., Echols, N., Headd, J. J., Hung, L.-W., Kapral, G. J., Grosse-Kunstleve, R. W., McCoy, A. J., Moriarty, N. W., Oeffner, R., Read, R. J., Richardson, D. C., Richardson, J. S., Terwilliger, T. C., and Zwart, P. H. (2010). Phenix: a comprehensive python-based system for macromolecular structure solution. *Acta Crystallographica Section D: Biological Crystallography*, 66(2):213–221.

[3] Blankenberg, D., Von Kuster, G., Coraor, N., Ananda, G., Lazarus, R., Mangan, M., Nekrutenko, A., and Taylor, J. (2010). Galaxy: a web-based genome analysis tool for experimentalists. *Current protocols in molecular biology*, 89(1):19–10.

[4] Boettiger, C., Chamberlain, S., Hart, E., and Ram, K. (2015). Building software, building community: lessons from the ropensci project. *Journal of Open Research Software*, 3(1).

[5] Brett, A., Croucher, M., Haines, R., Hettrick, S., Hetherington, J., Stillwell, M., and Wyatt, C. (2017). Research software engineers: State of the nation report 2017. *Zenodo. DOI*, 10.

[6] Caporaso, J. G., Kuczynski, J., Stombaugh, J., Bittinger, K., Bushman, F. D., Costello, E. K., Fierer, N., Peña, A. G., Goodrich, J. K., Gordon, J. I., Huttley, G. A., Kelley, S. T., Knights, D., Koenig, J. E., Ley, R. E., Lozupone, C. A., Mcdonald, D., Muegge, B. D., Pirrung, M., Reeder, J., Sevinsky, J. R., Turnbaugh, P. J., Walters, W. A., Widmann, J., Yatsunenko, T., Zaneveld, J., and Knight, R. (2010). Qiime allows analysis of high-throughput community sequencing data. *Nature Methods*, 7(5):335–6.

[7] Carr, A. (2019). The Chan Zuckerberg Initiative, personal communication.

[8] Chen, W. (2014). Memex.

[9] Cox, S. E. (2019). Lamont-Doherty Earth Observatory, personal communication.

[10] Crone, T. (2019). Lamont-Doherty Earth Observatory, personal communication.

[11] Díaz-García, C., González-Moreno, A., and Jose Sáez-Martínez, F. (2013). Gender diversity within r&d teams: Its impact on radicalness of innovation. *Innovation*, 15(2):149–160.

[12] Donoho, D. and Stodden, V. (2015). Reproducible research in the mathematical sciences. *The Princeton Companion to Applied Mathematics, Nicholas J. Higham, Mark R. Dennis, Paul Glendinning, Paul A. Martin, Fadil Santosa, and Jared Tanner, editors, Princeton University Press, Princeton, NJ, USA*, pages 916–925.

[13] Driscoll, M. (2019). Northwestern University, personal communication.

[14] Eghbal, N. (2018). Methodologies for measuring project health.

[15] Fenner, M. (2017). Bids receives sloan foundation grant to contribute to numpy development.

[16] Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. *Computing in science & engineering*, 9(3):90–95.

[17] Hurwitz, B. (2019). University of California, personal communication.

[18] Jones, E., Oliphant, T., and Peterson, P. (2014). SciPy: open source scientific tools for Python.

[19] Katz, D., Allen, G., Barba, L., Berg, D., Bik, H., Boettiger, C., Borgman, C., Brown, C., Buck, S., Burd, R., de Waard, A., Eve, M., Granger, B., Greenberg, J., Howe, A., Howe, B., Khanna, M., Killeen, T., Mayernik, M., McKiernan, E., Mentzel, C., Merchant, N., Niemeyer, K., Noren, L., Nusser, S., Reed, D., Seidel, E., Smith, M., Spies, J., Turk, M., Van Horn, J., and Walsh, J. (2018). The principles of tomorrow's university [version 1; referees: 2 approved]. *F1000Research*, 7(1926).

[20] Kim, S. (2019). Amazon, personal communication.

[21] Knepley, M. (2019). University at Buffalo, personal communication.

[22] Knight, R. (2019). Qiime, University of California San Diego, personal communication.

[23] Lowndes, J. S. S., Best, B. D., Scarborough, C., Afflerbach, J. C., Frazier, M. R., O'Hara, C. C., Jiang, N., and Halpern, B. S. (2017). Our path to better science in less time using open data science tools. *Nature ecology & evolution*, 1(6):0160.

[24] McKiernan, E. C., Bourne, P. E., Brown, C. T., Buck, S., Kenall, A., Lin, J., McDougall, D., Nosek, B. A., Ram, K., Soderberg, C. K., Spies, J. R., Thaney, K., Updegrove, A., Woo, K. H., and Yarkoni, T. (2016). Point of view: How open science helps researchers succeed. *eLife*, 5:e16800.

[25] McKinney, W. (2010). Data structures for statistical computing in python. In *Proceedings of the 9th Python in Science Conference*, volume 445, pages 51–56. Austin, TX.

[26] Millman, K. J. and Aivazis, M. (2011). Python for scientists and engineers. *Computing in Science & Engineering*, 13(2):9–12.

[27] Millman, K. J., Brett, M., Barnowski, R., and Poline, J.-B. (2018). Teaching computational reproducibility for neuroimaging. *Frontiers in Neuroscience*, 12:727.

[28] New York University, UC Berkeley, t. U. o. W. (2018). Creating institutional change in data science the moore-sloan data science environments.

[29] NSF (2018). Harnessing the data revolution (hdr): Data science corps (dsc).

[30] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in python. *J. Mach. Learn. Res.*, 12:2825–2830.

[31] Poldrack, R. A., Gorgolewski, K. J., and Varoquaux, G. (2018). Computational and informatics advances for reproducible data analysis in neuroimaging. *arXiv preprint arXiv:1809.10024*.

[32] Rupp, K. and Selberherr, S. (2011). The Economic Limit to Moore's Law. *IEEE Transactions on Semiconductor Manufacturing*, 24(1):1–4.

[33] Smith, A. M., Niemeyer, K. E., Katz, D. S., Barba, L. A., Githinji, G., Gymrek, M., Huff, K. D., Madan, C. R., Cabunoc Mayes, A., Moerman, K. M., Prins, P., Ram, K., Rokem, A., Teal, T. K., Valls Guimera, R., and Vanderplas, J. T. (2018). Journal of open source software (joss): design and first-year review. *PeerJ Computer Science*, 4:e147.

[34] Smith, N. and Vanderwalt, S. (2015). A better default colormap for matplotlib.

[35] Teal, T. K., Cranston, K. A., Lapp, H., White, E., Wilson, G., Ram, K., and Pawlik, A. (2015). Data carpentry: workshops to increase data literacy for researchers. *International Journal of Digital Curation*, 10(1):135–143.

[36] The Astropy Collaboration, Robitaille, Thomas P., Tollerud, Erik J., Greenfield, Perry, Droettboom, Michael, Bray, Erik, Aldcroft, Tom, Davis, Matt, Ginsburg, Adam, Price-Whelan, Adrian M., Kerzendorf, Wolfgang E., Conley, Alexander, Crighton, Neil, Barbary, Kyle, Muna, Demitri, Ferguson, Henry, Grollier, Frédéric, Parikh, Madhura M., Nair, Prasanth H., Günther, Hans M., Deil, Christoph, Woillez, Julien, Conseil, Simon, Kramer, Roban, Turner, James E. H., Singer, Leo, Fox, Ryan, Weaver, Benjamin A., Zabalza, Victor, Edwards, Zachary I., Azalee Bostroem, K., Burke, D. J., Casey, Andrew R., Crawford, Steven M., Dencheva, Nadia, Ely, Justin, Jenness, Tim, Labrie, Kathleen, Lim, Pey Lian, Pierfederici, Francesco, Pontzen, Andrew, Ptak, Andy, Refsdal, Brian, Servillat, Mathieu, and Streicher, Ole (2013). Astropy: A community python package for astronomy. *A&A*, 558:A33.

[37] Thomas, R. (2019). NERSC, personal communication.

[38] Tollerud, E. (2019). AstroPy, Space Telescope Science Institute, personal communication.

[39] Walt, S. v. d., Colbert, S. C., and Varoquaux, G. (2011). The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, 13(2):22–30.

[40] Walth, G. L., Wright, S. A., Rundquist, N.-E., Andersen, D., Chapin, E., Chisholm, E., Do, T., Dunn, J., Ellerbroek, B., Gillies, K., Hayano, Y., Johnson, C., Larkin, J., Nakamoto, T., Riddle, R., Simard, L., Smith, R., Suzuki, R., Sohn, J. M., Weber, R., Weiss, J., and Zhang, K. (2018). The infrared imaging spectrograph (iris) for tmt: advancing the data reduction system. volume 10707, pages 10707 – 10707 – 14.

[41] Zlotnick, F. (2017). Github open source survey 2017. `http://opensourcesurvey.org/2017/`.