NSF POSE: PHASE II: AN OPEN-SOURCE ECOSYSTEM FOR STATISTICAL PYTHON

K. JARROD MILLMAN, STÉFAN J. VAN DER WALT, AND SANDRINE DUDOIT
UNIVERSITY OF CALIFORNIA, BERKELEY

JONATHAN TAYLOR
STANFORD UNIVERSITY

IAIN CARMICHAEL
UNC CHAPEL HILL

NSF POSE: Phase II: An Open-Source Ecosystem for Statistical Python

Project Summary

Overview

Building on Phase I scoping and community discovery, this Phase II proposal will establish the Statistical Python Project (SPP) as the managing organization responsible for creating and growing a sustainable Open-Source Ecosystem (OSE) for statistical computing in Python. The ecosystem will unify scattered statistical research products into a coherent, discoverable, and trusted *domain-stack* (a collection of packages that address the needs of a specific domain), providing a robust foundation for applied statistics in research, education, and industry. SPP will formalize governance, strengthen contributor pathways, integrate infrastructure, and deliver training programs that empower educators, researchers, method developers, and practitioners. Implementations will be made in accordance with established community-vetted security practices, and progress will be evaluated using several quantitative and qualitative metrics, as well as through regular feedback from the community and an advisory board. By disseminating best practices in software engineering and creating a coherent entry point for statistics in Python, SPP will transform Python into a leading platform for statistical computing and education.

Intellectual Merit

The proposed Statistical Python OSE will advance knowledge by expanding the scope, interoperability, and reliability of statistical methodology in Python. It will improve access to cutting-edge statistical methods—including generalized linear models, regression, and modern inference techniques—through robust, community-owned software. The project will establish standardized APIs, shared infrastructure, and governance models, lowering barriers for statisticians and educators to contribute to and use the SPP domain-stack. By learning from the R ecosystem and aligning with other scientific Python domain-stacks, the OSE will accelerate dissemination of new statistical innovations and strengthen open-source research practices.

Broader Impacts

SPP will broaden participation by creating clear pathways for students, early-career researchers, and methodologists to become contributors and leaders in open-source software. Education will be advanced through reusable teaching resources, summer schools, and integration of statistical Python into classroom curricula. Industry collaboration will drive the adoption of robust statistical tools in applied workflows, while international partnerships will promote global accessibility and sustainability. The project will prepare and support the next generation of statisticians—through education, mentorship, and contributor pathways—while advancing reproducible, transparent, and impactful data science. In empowering a new generation of statisticians to develop and own their computational infrastructure, by providing a visible point of coordination, and by engaging with academia and industry, the SPP will form a sustainable OSE that enables cutting-edge statistics in Python and democratizes access to statistical methodological innovations.

Keywords: MPS; Statistics; Statistical Computing; Open-Source Software; Science Education

NSF POSE: Phase II: An Open-Source Ecosystem for Statistical Python

Project Description

1. **Overview.** The **Statistical Python Project (SPP)** [25] will serve as the managing organization responsible for creating and growing a sustainable, high-impact Open-Source Ecosystem (OSE) starting with two open-source research products. These two initial Python packages—developed by and for statisticians and researchers, each with small but active user and contributor communities—will seed a statistical *domain-stack* (a collection of packages that address the needs of a specific domain) and exemplify the types of projects we plan to integrate into the OSE during Phase II. Our vision is to catalyze a large-scale transformation in the statistical Python ecosystem, enabling both experienced and newly trained statisticians and researchers to contribute to a vibrant collection of interoperable, community-owned packages that make Python a robust and widely adopted platform for applied statistics.

Over the past year, with Phase I support, we conducted structured interviews with stakeholders across academia and industry, engaged with related domain-specific projects, participated in group discussions at national and international conferences, and convened a dedicated workshop to assess technical and social priorities. These activities confirmed clear opportunities to **unify, strengthen, and expand statistical tooling in Python** while lowering barriers for contributors and increasing adoption in education, research, and industry.

With Phase II support, we will move from planning to execution. Our work will formalize governance, strengthen and integrate infrastructure, expand contributor pathways, and deliver targeted training and outreach programs. Together, these efforts will transition statistical Python into an operational, sustainable, and community-driven ecosystem that fulfills NSF's Phase II objectives, supporting distributed, community-driven development and deployment of open-source tools in real-world environments.

1.1. *Context of OSE.* Scientific Python [13, 17] is a pillar of modern scientific computing, data analysis, and machine learning. It is composed of multiple domain-specific OSEs, each serving the needs of a particular research community.

These domain-specific OSEs all depend on a common set of *core* packages, such as NumPy (numerical computing) [32, 6] and SciPy (scientific algorithms) [31], which provide foundational data structures, algorithms, and tools. Each domain-specific OSE builds and maintains its own *domain-stack*: additional libraries, built on the core packages, that address the requirements of its discipline. For example, the PySAL project functions as the managing organization for the spatial analysis community, curating and supporting a cohesive stack of geospatial analysis libraries [28, 18]. Similarly, the Scikit-HEP project serves as the managing organization for particle physics, coordinating its own stack of HEP-specific tools—used and developed by high-energy particle (HEP) physicists [29, 19].

These domain-stacks are actively developed and maintained by their respective scientific communities, and are used both for research and teaching. In industry, they also often form part of the daily toolset used by domain practitioners. They are typically organized in layers, starting from the shared general-purpose scientific infrastructure and extending toward increasingly specialized methods, workflows, and discipline-specific applications.

The Scientific Python Project [23], founded by PI Millman and Co-PI van der Walt, coordinates the entire ecosystem and supports the community of contributors and maintainers. In particular, the project provides ecosystem-wide infrastructure such as the Scientific Python SPECs (community standards and specifications) that ensure consistency and interoperability across domain-stacks [21]. It also develops widely used educational and training resources, including the Scientific Python Lecture Notes [22], which serve as a shared foundation to introduce students and researchers to scientific computing with Python. These community standards and educational resources underpin our proposed work and provide essential resources for transitioning new projects into the Statistical Python OSE.

Statistical Python. Python is widely adopted in data science, and its use for statistics is expanding rapidly—particularly in education and applied research. The statistical ecosystem in Python is currently anchored by six core libraries:

- NumPy, which provides fast, flexible array and numerical operations, and underpins nearly all statistical and scientific computing in Python. It supports descriptive statistics, correlation and covariance computations, random sampling, and tools for constructing histograms and binning data.
- pandas [12, 30], which offers intuitive, high-performance data structures for tabular and time series data, making data cleaning, wrangling, reshaping, aggregation, and exploratory analysis straightforward and efficient.
- SciPy, which builds on NumPy to deliver a broad range of scientific and statistical functionality—including, in its `scipy.stats` submodule, a comprehensive suite of probability distributions, summary statistics, and basic statistical tests. It also provides modules for clustering, optimization, interpolation, and signal processing.
- Matplotlib [8], the foundational plotting library in Python, which enables the creation of high-quality static, animated, and interactive visualizations, and serves as the basis for many higher-level plotting and statistical graphics libraries.
- statsmodels [24], which offers tools for econometrics, classical statistics, and statistical modeling—including linear and generalized linear models, time series analysis, survival analysis, and hypothesis testing, with extensive support for model diagnostics and statistical inference.
- scikit-learn [16], which is best known for machine learning but also supports statistical modeling, offering a consistent API for regression, classification, clustering, model evaluation, statistical preprocessing, and dimensionality reduction.

These core libraries are well-tested, reliable, and uphold high software engineering standards, making them trusted foundations for research and application. They benefit from contributions by scientific users, methods developers, and software engineers. Libraries like scikit-learn are especially valued for their clean, consistent interfaces and their integration with the broader Python data stack.

In addition to these core libraries, there is a large and growing number of smaller, specialized open-source research products, often written by statisticians, such as *yaglm* [1], *islp* [27], *pygam* [26], *formulaic* [33], and *seaborn* [34]. These specialized packages provide

advanced statistical functionality, modeling approaches, and domain-specific tools. For example, *yaglm* offers state-of-the-art implementations of generalized linear models with advanced regularization techniques, while *islp* supports interactive educational resources tied to foundational statistical learning materials. *Pygam* enables flexible generalized additive modeling for capturing complex nonlinear relationships in data, and *formulaic* provides a convenient formula interface for statistical model specification, expanding interoperability among modeling libraries. *Seaborn* enhances data visualization by building on Matplotlib, delivering attractive and informative graphics tailored for statistical analysis.

However, while these smaller packages add valuable functionality to the ecosystem, they tend to be less well-known and therefore less widely used than the core libraries. Many have limited testing and documentation, do not consistently follow established software engineering standards, and often have small developer communities—sometimes even a single maintainer. As a result, their long-term sustainability, discoverability, and interoperability within the broader statistical Python ecosystem can be uncertain. Following the example of successful domain-specific projects such as PySAL and Scikit-HEP, SPP will address these challenges by establishing and curating a coherent domain-stack—a coordinated set of packages tailored to statistical computing—built on the shared foundations of the broader scientific Python ecosystem.

Comparison to R. R remains the gold standard for statistics, benefiting from strong branding, an integrated and cohesive ecosystem, and an extensive array of teaching resources. R's tidyverse and RStudio together provide a seamless and unified experience, supporting clear syntax, consistent data paradigms, and rapid exploration. The central CRAN repository facilitates the broad dissemination of thousands of statistical packages, ranging from high-quality, production-level libraries to research prototypes and packages that support course materials. This flexibility encourages innovation and lowers the barrier for method developers to share new techniques with the wider community. Within R, subcommunities like the tidyverse and Bioconductor have established additional governance, developer guidance, documentation standards, and release infrastructure that further advance reproducibility and software quality.

Unlike Python—which is a general-purpose programming language with strengths in fields from web development to scientific computing—R was specifically designed for statistics, making it the historical default in academic statistics departments. As a result, the R ecosystem continues to be driven by contributions from method-focused statisticians and serves as the primary medium for teaching and disseminating new statistical techniques. However, this tradition is evolving: for example, last year the graduate-level computational statistics course at UC Berkeley was taught in Python for the first time, based on consistent feedback from graduate students in prior years seeking greater alignment with broader data science trends and industry demand. A summary comparison between Python and R is shown in Table 1.

Weaknesses and Needs. Despite Python's growing prominence in data science and scientific computing, its statistics ecosystem remains fragmented and challenging to navigate, with critical gaps in interoperability, usability, contributor pathways, and educational resources—especially when compared to R's more unified community and infrastructure.

Aspect	**Python (Scientific Python)**	**R (CRAN, tidyverse)**
Experience	Fragmented, less cohesive	Integrated and cohesive
Teaching	Improving, but less abundant	Extensive, beginner-friendly
Community	Large, but less statistics-focused	Strong, statistics-focused
Packaging	High barriers, less modularity	Easy, many small packages
Interoperability	Incompatible data structures, APIs	Strong within tidyverse, RStudio
Branding	Data science/machine learning focus	Statistics focus

TABLE 1. *Summary Comparison of Python and R for Statistics.*

Key barriers, identified during Phase I, include:

- **(KB1) Absence of a unified entry point:** Unlike R's tidyverse, or Base R's data frames, Python lacks a coherent, ubiquitous framework for statistical computing.
- **(KB2) Diffuse community identity:** The Python statistics community lacks the cohesion, leadership structures, and recurring events that characterize R's statistics community.
- **(KB3) High barriers to contribution:** Strict standards and tightly coupled codebases deter new contributors; many small, innovative packages remain invisible and underused in an ecosystem that tends to aggregate functionality into larger, well-maintained libraries.
- **(KB4) Limited teaching resources:** There are far fewer beginner-friendly, statistics-focused tutorials, examples, and case studies than in R's ecosystem.
- **(KB5) Fragmentation:** Core libraries (e.g., *statsmodels*, *scikit-learn*) employ incompatible APIs and workflows, creating a steep learning curve for users and educators.
- **(KB6) Interoperability gaps:** Core data structures (*pandas*, *NumPy*) are not consistently supported across statistical libraries, requiring manual conversions and producing inconsistent outputs.
- **(KB7) Incomplete statistical coverage:** Support for foundational statistical methods is uneven, and advanced or niche techniques often lag far behind R's *CRAN* offerings.
- **(KB8) Lack of comprehensive diagnostics and communication tools:** Many libraries omit established numerical diagnostics, visualizations, and summary reports—capabilities common in other statistical ecosystems.
- **(KB9) Tight coupling of computation and methodology:** Statistical implementations often hard-code a single algorithm, limiting flexibility to choose alternatives better suited to different data sizes, hardware environments, and performance requirements.

These limitations constrain the dissemination and adoption of new statistical methods, hinder contributions from educators and methodologists, and reduce Python's effectiveness as a platform for reproducible research. Addressing these gaps—through more consistent

APIs, better integration, richer educational materials, and a visible community identity—would position Python as a true peer to R, while leveraging its existing strengths in data science and machine learning.

Long-term Vision and Guiding Principles. SPP aims to unify and strengthen statistical computing in Python by creating a vibrant, self-sustaining OSE that is deeply interconnected with the broader Scientific Python and statistics communities. This ecosystem will enable statisticians, educators, researchers, and practitioners to develop, share, and use statistical tools with ease, confidence, and measurable impact.

Our vision is of a community-owned platform that adopts and disseminates best practices in software engineering, governance, and documentation, while driving improvements in usability, interoperability, and discoverability across statistical packages.

SPP's mission is to empower educators with a free, coherent ecosystem for teaching statistics; to enable researchers to produce reliable, reproducible analyses using robust, well-engineered libraries; to provide method developers with standardized, interoperable pathways for swiftly delivering new statistical innovations to a wide audience; and to equip practitioners with powerful, efficient workflows that remove unnecessary friction between statistical and data science tools.

Societal Needs and Broader Impact. Statistical software is a fundamental component of nearly all research software stacks. It is essential that algorithmic advancements are made available as widely as possible, in the form of robust, well-tested, and well-maintained implementations. This proposal aims to advance that goal by fostering a two-directional exchange: disseminating best software engineering practices to statisticians, while bringing the methodological expertise of statisticians into research software development in Python.

The OSE will aggregate knowledge on how to transition experimental research code into robust, high-quality research software, creating a centralized resource that can also be used by other OSEs. By hosting fundamental educational resources on statistical techniques and propagating software development best practices, the OSE will provide clear pathways for onboarding academics into statistical scientific software development.

This approach will particularly empower early-career researchers, enriching statistical research and research software, while simultaneously expanding and revitalizing the open-source contributor community.

1.2. *Ecosystem Establishment and Growth.* In Phase I, we centered our exploratory work on two pilot projects that exemplify both the opportunities and challenges of integrating statistical methodology into the broader scientific Python ecosystem.

- **ISLP** [27]: The companion Python package for the textbook *An Introduction to Statistical Learning with Applications in Python*, closely following the examples in its widely used R version. Beyond providing computational workbooks, ISLP includes Pythonic design-matrix construction, a simple implementation of Bayesian Additive Regression Trees, and object-oriented stepwise model selection tools—making it directly useful in educational and applied research contexts.
- **YAGLM** [1]: A Python package for generalized linear models (GLM), offering modern statistical methodology with a wide range of loss functions, regularizers,

and solvers. GLMs are a flexible and powerful generalization of ordinary linear regression, encompassing many statistical models widely used in diverse applications. Beyond basic LASSO and ridge penalties, YAGLM supports structured penalties such as the nuclear norm as well as group, exclusive, fused, and generalized lasso, along with non-convex penalties. It provides a variety of tuning parameter selection methods, including cross-validation, information criteria with favorable model selection properties, and degrees-of-freedom estimators.

During Phase I scoping, we also identified and engaged with other emerging projects that could become valuable members of the Statistical Python domain-stack.

Current Organizational Foundation. To prepare for ecosystem growth, we established the **SPP GitHub Organization** [25] as the central hub for code repositories, governance documents, community guidelines, and coordination of development activities. This hub, along with our website, will serve as the primary gateway for users to discover ecosystem projects, access documentation, and become contributors themselves. In Phase II, we will formalize governance, define clear criteria for recognizing a project as an SPP package, establish review and approval processes for candidate projects, and coordinate activities between core maintainers, affiliated projects, and new community initiatives.

Ecosystem Establishment Strategy. Our ecosystem establishment plan focuses on structured recruitment, onboarding, and integration of high-quality libraries already used by the statistical and scientific Python communities. We will begin by identifying candidate packages through GitHub, PyPI, and citation networks, searching for actively maintained projects central to statistical computing. We will also solicit recommendations from Phase I workshop participants, academic collaborators, and industry partners, and leverage connections within related domain-specific OSEs such as Scientific Python, scikit-HEP, and PySAL to discover projects with overlapping needs. Candidate projects will then be evaluated against SPP's quality criteria, taking into account factors such as maintenance activity, test coverage, documentation quality, interoperability with core data structures, and governance maturity. Once suitable projects are identified, we will support their adoption and integration into the ecosystem by providing technical assistance with packaging, continuous integration, documentation building, and release management. We will also offer governance and community management templates adapted from established OSEs, and help align APIs and naming conventions across libraries to reduce usage friction. A unified SPP website and documentation portal will serve as a discoverability pathway, providing an indexed registry of certified ecosystem packages.

Ongoing Discovery and Engagement. Ecosystem growth will be supported by sustained discovery and engagement throughout the life of the project. We will maintain a recurring landscape review to identify emerging statistical methods libraries on GitHub and PyPI, and will conduct direct outreach to maintainers to discuss possible affiliation and support needs. Engagement will be fostered through Birds-of-a-Feather sessions, tutorials, and poster presentations at community events such as SciPy, PyData, and the Joint Statistical Meetings. User and developer training will bring in graduate students, early-career

researchers, and method developers, while open community calls and public discussion forums will be used to collect and address feedback from both users and contributors.

Industrial and International Collaboration. Engagement beyond academia will be essential for both impact and sustainability. We will build industrial partnerships by leveraging our Phase I connections with technology firms and data-driven companies to identify priority features and scalability requirements, co-develop case studies and training materials based on real-world workflows, and explore co-sponsorship of internships, sprints, and hackathons. In parallel, we will foster deep engagement with the global statistics community by actively participating in international conferences, collaborative projects, and standards development, and by inviting statisticians worldwide to contribute to the ecosystem. We will further strengthen our ties to the scikit-learn team in France, coordinating joint workshops, technical exchanges, and governance discussions to ensure interoperability and best practice dissemination across the Statistical Python OSE.

Long-Term Growth Objectives. Through this coordinated program of establishment and discovery, SPP will create a clearly branded, discoverable, and trusted ecosystem for statistical computing in Python. It will provide method developers with a low-friction path from novel algorithm to widely adopted package, and will serve educators and students with curated, well-documented tools for teaching statistics. Researchers and industry practitioners will be able to combine statistical methods seamlessly with the broader scientific Python stack. A globally networked contributor community will emerge, capable of sustaining and evolving the ecosystem well beyond the life of the NSF award.

1.3. *Qualification of the Team.* As outlined in the Intellectual Merit section, this is an ambitious proposal with significant potential impact. It calls for extensive experience in statistics, software development, and open-source community management. Our team is exceptionally well qualified in all of these areas, with members serving as founding contributors to Scientific Python and earning recognition as leaders in the creation, governance, and sustainability of open-source statistical and scientific software. UC Berkeley's and Stanford's Statistics departments are global leaders in research and education, while UC Berkeley's College of Computing, Data Science, and Society and UNC Chapel Hill's School of Data Science and Society emphasize translating discovery into real-world impact across the nation and the world. Our team is therefore uniquely qualified to execute the proposed activities, backed by the institutional resources and reputation needed for success.

PI Millman, Executive Director of UC Berkeley's Open Source Program Office (OSPO), brings over two decades of leadership in scientific Python development, governance, and community sustainability. He co-founded NumFOCUS [15], served as NumPy and SciPy Release Manager (2007–2009) and on the SciPy Steering Committee (2007–2010), and was an early contributor to scikit-learn. Millman co-founded the Neuroimaging in Python (NIPY) project and the Scientific Python project, both of which established cross-domain best practices for open and reproducible research. He currently serves as release manager for multiple foundational libraries, including NetworkX, scikit-image, pygraphviz, and numpydoc, and has extensive experience building governance and infrastructure for research software communities.

Co-PI van der Walt, Senior Research Data Scientist at the Berkeley Institute for Data Science (BIDS), is the founder of scikit-image, co-author of *Elegant SciPy*, and software architect of the SkyPortal astronomy data platform. He serves on the NumPy Steering Committee and is a NumFOCUS Emeritus Director. His contributions span core ecosystem projects such as NumPy, SciPy, scikit-image, and NetworkX, where he has focused on API design, documentation standards, and governance models that have strengthened scientific Python's long-term sustainability. He is a co-founder of the Scientific Python project and has extensive experience coordinating multi-institution open-source collaborations.

Co-PI Dudoit, Executive Associate Dean, College of Computing, Data Science, and Society and Professor of Statistics and Public Health at UC Berkeley, has extensive experience developing open-source statistical software, including as a founding core developer of the Bioconductor project—one of the largest and most successful open-source ecosystems in computational biology. Her research bridges statistical methodology, computational biology, and high-dimensional data, and she has played a leading role in integrating rigorous statistical standards into collaborative, cross-disciplinary software communities.

Sustainability Lead Whitaker, Executive Director of BIDS, is recognized for building collaborative open-source communities. She founded *The Turing Way*, an open handbook on reproducible, ethical, collaborative data science, and previously directed Tools, Practices & Systems research at the Alan Turing Institute.

Co-PI Taylor, Professor of Statistics at Stanford, specializes in selective inference and signal detection in structured noise. A long-time contributor to the scientific Python community and collaborator with the R community, he co-founded the Neuroimaging in Python project. In partnership with Trevor Hastie, he is lead developer of `ISLP`, the Python companion to the highly influential text *An Introduction to Statistical Learning* [10], bridging statistical education between the R and Python ecosystems.

Co-PI Carmichael, Assistant Professor of Pathology and Data Science at UNC-Chapel Hill, develops statistical and machine learning algorithms for complex data including networks, images, and multi-view data. He is the creator of `YAGLM`, an advanced Python package for generalized linear models with modern penalties and solvers. His work couples state-of-the-art statistical methodology with open-source best practices, directly shaping how the ecosystem supports robust, reproducible method implementations.

2. **Organization and Governance.** SPP will serve as the managing organization for the Statistical Python OSE, following governance and operational models adapted from successful domain-specific scientific Python communities like PySAL and scikit-HEP. All software produced under SPP will be released under a permissive MIT or BSD 3-Clause License. Documentation, tutorials, and learning resources will be released under the Creative Commons CC-BY license, promoting circulation, adaptation, and attribution.

The organizational structure will consists of community members and contributors, dedicated project developer teams for individual libraries or tools, a Steering Council (SC), and a Project Lead (PL). The SC will be responsible for guiding the overall scope, vision, and direction of the project, including the meta-package and affiliated packages. It will set strategic priorities and manages collaborations with other organizations or individuals. It will also manage the services operated by the project for the benefit of the community,

resolve issues when community discussion fails to reach consensus, and update project policies as needed. Project developer teams will maintain autonomy in their internal processes but align with the broader governance principles of SPP.

SPP will follow a **consensus-seeking decision-making process** modeled after the governance approaches used by NetworkX, scikit-image, and other mature scientific Python projects. All substantive project management discussions will occur on public community channels, such as discussion forums, issue trackers, and mailing lists, except for sensitive matters that may require private discussion. Decisions will be made by seeking a resolution with no open objections from the SPP community. If consensus cannot be reached within a reasonable timeframe, the decision will be escalated to the SC. The SC will attempt to resolve the matter through further consensus, and if deadlocked, a simple majority vote of the SC will determine the outcome. The PL will hold final decision-making authority. In practice, the PL will defer this authority to the consensus of the community and the SC, intervening only when necessary.

The SPP will be formally launched and established with Phase II support. The SC will initially comprise the principal investigators and senior personnel from this NSF award, with PI Millman serving as the initial PL. This group will meet monthly to review progress, address challenges, and ensure that activities stay aligned with community interests, project milestones, and best practices.

To broaden input and ensure the ecosystem aligns with national and global trends, an Advisory Board—consisting of Bin Yu (Professor of Statistics, UC Berkeley), Genevera Allen (Professor of Statistics, Columbia University), Gordon Watts (Professor of Physics, University of Washington), Trevor Hastie (Professor of Statistics, Stanford University), and Gaël Varoquaux (Research Director, INRIA)—will convene quarterly to provide strategic guidance. This board ensures responsiveness and cross-pollination with leading domain communities.

3. **Continuous Development Model.** SPP will use the SPP GitHub Organization as the central platform for code hosting, collaboration, project management, and documentation. This will provide a single, discoverable namespace for all ecosystem projects, ensuring transparency, open access, and a clear entry point for contributors, users, and partners.

Our continuous development model will follow best practices documented by the Scientific Python ecosystem specifications [21] and related resources, including guidelines for packaging, testing, documentation, accessibility, and governance. The SC will finalize the specific inclusion criteria for SPP-affiliated projects, but these criteria will emphasize consistent use of community standards, reproducible workflows, maintainable code, and robust documentation.

All projects will adopt shared, open development practices. This includes GitHub-based version control for collaborative and asynchronous code contributions, issue tracking for transparent task and defect management, and adherence to the SPP Code of Conduct. Each repository will include a detailed contributor guide, documented governance processes aligned with the overall SPP governance model, a code review process, a public development roadmap, and an enhancement proposal process for major changes.

Projects in the ecosystem will implement continuous integration and deployment (CI/CD) pipelines using GitHub Actions. Automated tests, linting, and style checks will run for every pull request to ensure code quality before merge.

Releases will be made on a regular and predictable schedule, with stable versions published to the Python Package Index (PyPI) using secure, automated GitHub workflows protected by two-factor authentication. Each project will maintain a changelog to ensure users and downstream developers are aware of updates and their implications.

Documentation will be built automatically using Sphinx or similar tools and deployed on each release to a public, versioned documentation site. Projects will follow the Scientific Python "repo review" guidelines [20] to ensure documentation, testing infrastructure, and governance metadata are complete and discoverable.

4. **Risk Analysis/Security Plan.** The Statistical Python OSE faces similar security risks as the rest of the ecosystem, including supply chain vulnerabilities, access control weaknesses, and insider threats due to a distributed contributor base. Supply chain issues arise from reliance on core Scientific Python dependencies—for example, when specialized statistical packages like YAGLM inherit upstream vulnerabilities that can then cascade throughout the ecosystem. Access control risks arise from managing permissions across multiple SPP GitHub repositories, as overprivileged or stale credentials could permit unauthorized changes. The difficulty in fully vetting every new contributor in an open ecosystem increases the risk that vulnerabilities may be introduced, whether deliberately or inadvertently, through seemingly legitimate code submissions.

To address these, SPP will adopt a security framework aligned with SPEC standards: enforce multi-factor authentication (MFA) for all maintainers, grant only minimal permissions to contributors, and use GitHub's automated security scanning, code quality checks, and cryptographic signing (SLSA Level 2) for releases.

Both technical safeguards and strong community protocols are envisioned: we will form a security response team of trained maintainers to manage vulnerability disclosures and remediation, and no sensitive or personally identifiable data will be stored in repositories.

SPP will consult with advisors such as Seth Larson (Python Software Foundation's Security Developer-in-Residence) and Dustin Ingram (Google's Open Source Security Team and PyPI maintainer), and actively engage with the Scientific Python security working group to implement best security practices. Security training will be integrated into contributor onboarding and bootcamps, establishing a community-driven, sustainable approach that advances security across the Statistical Python and wider Scientific Python ecosystems.

5. **Community Building.** Our vision is to cultivate a welcoming, mission-driven community that inspires sustained contributions from students, educators, researchers, and method developers. Based on Phase I interviews and workshop feedback, the timing is ideal to invest in Statistical Python community building. Incoming statistics students and new faculty increasingly use Python, and many are explicitly requesting statistics training and contribution pathways. We will meet this demand through coordinated efforts across software development, educational programming, and community engagement, structured as a two-year progression from establishment to expansion. The primary focus of Year 1 will be to establish SPP infrastructure and governance, strengthen the initial pilot projects,

and develop educational programs. In Year 2, the focus will shift to expanding the domain-stack, launching educational programs, and broadening community engagement.

Year 1: Foundation and Establishment. Community organization efforts in Year 1 will include setting up the SPP governance structure and forming an initial SC. The SC will hold two planning meetings with the grant team and will formalize criteria for including projects in the SPP domain-stack. Drawing on our collective expertise and Phase I findings, the SC will also begin drafting a long-term SPP roadmap. We will conduct broad outreach to identify promising projects, launch monthly community calls, deploy a discussion forum, and expand the central landing page. Satellite events and Birds of a Feather sessions will be organized at relevant conferences.

We will bring the pilot packages, YAGLM and ISLP, into alignment with software engineering best practices established in the broader Scientific Python ecosystem. This will include implementing comprehensive testing and code coverage, robust CI/CD pipelines, and well-defined package governance. Following recommendations from Scientific Python SPECs, we will introduce secure release automation and cross-project testing.

On the educational front, we will design a five-day intensive developer bootcamp, *How to Contribute to Statistical Python*, and pilot it as a sequence of weekly sessions at UC Berkeley, leveraging integration with existing courses. We will simultaneously develop a comprehensive two-week *Python for Statisticians* curriculum, building on our team's existing materials and drawing from open educational resources such as the Scientific Python Lecture Notes.

Year 2: Growth and Expansion. At least five additional projects will be onboarded, each supported to adopt the software engineering practices and governance structures established in Year 1. We will help automate releases, support cross-project testing, and produce contributor guides, enhancement proposal templates, and reference CI/CD configurations.

Our educational program will expand with the launch of the two-week *Python for Statisticians* summer school at UC Berkeley, offered at a modest fee to cover costs and test a sustainable model. We will also deliver two developer bootcamps—one at UC Berkeley and the other at UNC—open by application to participants with basic Python proficiency and a proposed area of contribution.

To expand community engagement, we will deploy an online forum for Q&A and topic-specific discussion, and establish a dedicated community manager to coordinate issue triage, onboarding, and developer sprints. Regular hackathons and sprints will be scheduled with clear goals and mentorship opportunities, complemented by an expanded presence at major conferences including SciPy, PyData, and JSM. Throughout Year 2, we will engage the broader SPP community in an open process to further develop and refine our long-term roadmap for the project.

6. **Sustainability.** By integrating proven open-source community practices with diversified revenue streams and broad partnership networks, SPP will provide the durable organizational and financial infrastructure needed for the Statistical Python OSE to thrive and meet the evolving needs of students, educators, researchers, and practitioners worldwide.

Community Sustainability. We will support contributors at all stages of the open-source "mountain of engagement" [14]. The SPP ecosystem will include coherent discovery pathways so that users can learn about the project, make first contact, and participate through clear contributing guidelines and regular onboarding calls. We will host monthly Collaboration Cafes [3, 4]. These co-working calls, developed by *The Turing Way*, foster opportunities for sustained and networked participation. Contributors will be encouraged to identify new ways to deepen collaboration, including inviting colleagues and partners to engage in the ecosystem. By the end of the grant, we will have transitioned to a leadership model that does not rely on a single person, institution, or employer. Notes from SC and Advisory Board meetings will be made publicly available, and community input will be incorporated into decision-making at all stages. Ahead of each meeting we will invite community feedback through public forums, while also providing opportunities for confidential or anonymous submissions by email. We will pair these feedback loops with activities to celebrate and acknowledge contributions in ways that best support contributors' career and personal needs, following *The Turing Way's* contributor record [5]. This will include promoting contributions through blog posts and social media, and providing opportunities for contributors to represent SPP at conferences and other events.

Financial Sustainability. To ensure long-term viability, SPP will pursue a diversified funding model that blends institutional support, industry partnerships, program revenue, and grant funding. This approach reduces reliance on any single funding source and aligns community and stakeholder interests.

- *Institutional Support Networks.* SPP will operate within BIDS and the UC Berkeley OSPO. BIDS is a member of the Academic Data Science Alliance and the Community for University and Research Institution OSPOs. The Berkeley OSPO is part of the University of California OSPO network, a collective of six OSPOs working together to amplify the role of open source in research, public service, and education. We will also collaborate with Stanford's OSPO and its partnership with PyOpenSci. SPP will apply learnings from these open-source networks to secure ongoing financial and in-kind support, including faculty release time, student internships, infrastructure access, and hosting of educational programs and community events.
- *Industry Partnership Development.* SPP will leverage BIDS membership in the Berkeley Corporate Engagement Network to expand industry partnerships across the SPP ecosystem. Sustainability will include financial support for core maintenance and sponsorship of developer bootcamps, community meetings, and infrastructure costs. Industry partners will benefit from early visibility into emerging methods, funding bug fixes and feature requests, and access to expert developer advice.
- *Revenue-Generating Programs.* The 2-week "Python for Statisticians" summer school, piloted in Year 2, will establish a fee-for-service model that supports operations, instructor compensation, and infrastructure maintenance. This program will serve as a template for additional educational offerings and workshop series that generate revenue while advancing the mission.

- *Grant Portfolio Strategy.* We will pursue targeted development grants from foundations, government agencies, and industry sponsors to support package maintenance, feature development, and ecosystem infrastructure.

7. **Evaluation plan.** Our evaluation philosophy aligns with NSF's POSE Phase II goals: enabling distributed adoption, sustained growth, and measurable impact of open-source ecosystems. We view metrics not as static counts but as signals of whether SPP is lowering barriers to contribution, fostering self-sustaining communities, and driving adoption across research, education, and industry.

Ecosystem Development, Adoption, and Scholarly Impact. Using developer insight dashboards developed by the Scientific Python project, we will track community health across SPP projects. We will observe usage growth, and characterize the adoption of common tooling and Scientific Python recommended best practices across the SPP—a proxy for reduced maintainer burden. Through the Scientific Python community, educational events, and at conferences, we will invite users to share their experiences with the projects in the ecosystem, focusing on aspects like documentation quality, cross-library compatibility, data structure support, and reduced workflow friction. Similarly, we will ask contributors to the ecosystem to provide us with regular feedback via our online forum, at conferences, as well as at our developer meetings, focusing on aspects such as barriers to contribution, usability of developer documentation, the availability of cross-project tooling, as well as convergence towards unified APIs and common data structures. In terms of OSE growth, a clear indicator is the number of officially designated and supported SPP projects. For those, usage evidence will include PyPI and conda-forge download counts as well as GitHub stars and forks. Following the Chan Zuckerberg Initiative's practices, we will also track citations of these libraries in the scholarly literature [7, 9]. To supplement quantitative metrics, we will invite community members—including event participants—to share examples of where SPP tools enabled research breakthroughs, industry applications, or policy insights. In contrast to these finer-grained metrics, the advisory board will help assess progress on higher-level initiatives and suggest strategic course alterations.

Community Growth and Engagement. We will measure contributor growth by tracking the number and diversity of contributors, including their geographic spread, academic affiliation, and progression through contributor, maintainer, and leadership roles. Participation in community calls, online forums, workshops, sprints, and bootcamps will serve as direct indicators of engagement. Onboarding pathways will be examined by recording the number of contributors recruited via workshops and their progression into long-term contributor or maintainer roles.

Education and Outreach. Educational impact will be measured through participation rates in bootcamps, summer schools, and online tutorials. Following The Carpentries model [11], we will collect evaluation data from students and instructors using pre- and post-assessment surveys [2]. A key metric will be the reuse of SPP training materials, including the number of institutions and courses adopting them and the extent of reuse of notebooks, slides, datasets, and recordings. We will encourage a community-engaged approach by

inviting instructors to reuse evaluation surveys in their classrooms and by providing direct pathways for learners to join the SPP community.

8. **Intellectual Merit.** This project will advance knowledge by establishing the first coordinated, community-owned OSE for statistics in Python. Importantly, the OSE addresses a persistent gap: while the scientific Python ecosystem is robust and widely adopted, statisticians—who generate many methodological innovations—remain underrepresented, and Python users lack cohesive access to modern statistical methodology. By bringing established projects such as YAGLM (for generalized linear models) and ISLP (for statistics education) under a single umbrella, and by aligning their development with best practices from the Scientific Python community, SPP will deliver a sustainable framework for providing advanced, reproducible statistical methods through community-driven open-source software. Establishing standardized APIs, governance models, and interoperability conventions will further reduce barriers to participation and enhance reproducibility across disciplines. These activities will empower researchers to apply cutting-edge inference, regression, and regularization techniques more broadly, while also modeling how collaborations between statisticians and computing communities can succeed and scale. More broadly, SPP's impact extends beyond Python by generating insights into how research code can become durable infrastructure—an open question of broad relevance to statistics and many other methodologically driven domains. Finally, the project will reinforce education and workforce development by tightly integrating research and training. Bootcamps, summer schools, and open curricula will equip students, educators, and early-career researchers to both apply statistical methods and practice sustainable open-source development. This dual emphasis on methodology and engineering is a novel strategy to foster contributors capable of both innovating and sustaining research software. Together, these advances will help Python become a true peer to R for statistics, while yielding transferable lessons at the intersection of statistics, software engineering, and open-source governance.

Broader Impacts

SPP will broaden participation in open-source software and support the growth and advancement of statistical professionals by offering clear contributor pathways for students, early-career researchers, and method developers. By lowering barriers to entry through standardization, improved infrastructure, and modern communication platforms, SPP will foster a welcoming environment that supports progression from user, to contributor, to maintainer, and ultimately to leader. This approach will ensure continuity and enduring community stewardship well beyond the NSF funding period. The project will provide direct educational impact through the development of teaching resources, developer bootcamps, and a two-week "Python for Statisticians" summer school. These programs will serve as powerful training grounds for contributors, while also acting as scalable, fee-supported models for long-term sustainability. By bringing tools like ISLP into the classroom, SPP will provide instructors with a coherent Python-based alternative to R, meeting the evolving needs of data science education and workforce preparation. Students

will acquire not only statistical knowledge but also essential skills in reproducibility, collaboration, and open-source development—capacities that are increasingly critical in both research and industry. SPP's impact will reach beyond academia, supporting cross-sector and international partnerships that reinforce the ecosystem's sustainability. Industry partners will benefit from improved, robust statistical tools, while international collaborations will advance global alignment of open-source practices and standards. In a broader sense, society will benefit from more transparent, reproducible, and accessible statistical methods, ultimately enabling reliable, evidence-based decision making in key areas such as health, education, and policy.

References

[1] Iain Carmichael, Thomas Keefe, Naomi Giertych, and Jonathan P. Williams. *yaglm: A Python package for penalized generalized linear models that supports fitting and model selection for structured, adaptive and non-convex penalties*. https://github.com/statistical-python/yaglm. Preliminary release. Actively developed as part of the Statistical Python Project. 2021.

[2] *Carpentries Assesment Archives*. URL: https://carpentries.github.io/assessment-archives/.

[3] *Co-working*. URL: https://book.the-turing-way.org/collaboration/coworking.

[4] *Collaboration Cafes*. URL: https://book.the-turing-way.org/community-handbook/community-calls/community-calls-collabcafe.

[5] *Contributor Record*. URL: https://book.the-turing-way.org/community-handbook/acknowledgement/acknowledgement-record.

[6] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. "Array programming with NumPy". In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.

[7] Kate Hertweck, Carly Strasser, and Dario Taraborelli. *Insights and Impact From Five Cycles of Essential Open Source Software for Science*. July 2024. DOI: 10.5281/zenodo.11201216. URL: https://doi.org/10.5281/zenodo.11201216.

[8] John D Hunter. "Matplotlib: A 2D graphics environment". In: *Computing in Science & Engineering* 9.3 (2007), pp. 90–95. URL: https://doi.org/10.1109/MCSE.2007.55.

[9] Ana-Maria Istrate, Donghui Li, Dario Taraborelli, Michaela Torkar, Boris Veytsman, and Ivana Williams. *A large dataset of software mentions in the biomedical literature*. 2022. arXiv: 2209.00693 [cs.DL]. URL: https://arxiv.org/abs/2209.00693.

[10] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning: with Applications in R*. Vol. 112. Springer, 2013.

[11] Kari Jordan, François Michonneau, and Belinda Weaver. *Analysis of Software and Data Carpentry's Pre- and Post-Workshop Surveys*. July 2018. DOI: 10.5281/zenodo.1325464. URL: https://doi.org/10.5281/zenodo.1325464.

[12] Wes McKinney. "Data Structures for Statistical Computing in Python". In: *Proceedings of the 9th Python in Science Conference*. Ed. by Stéfan van der Walt and Jarrod Millman. 2010, pp. 56 –61. DOI: 10.25080/Majora-92bf1922-00a.

[13] K Jarrod Millman and Michael Aivazis. "Python for scientists and engineers". In: *Computing in science & engineering* 13.2 (2011), pp. 9–12.

[14] *Mount of Engagement Blogpost*. URL: https://github.blog/open-source/maintainers/who-will-maintain-the-future-rethinking-open-source-leadership-for-a-new-generation/.

[15] *NumFOCUS*. URL: https://numfocus.org.

[16] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12.Oct (2011), pp. 2825–2830. URL: http://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf.

[17] Fernando Perez, Brian E Granger, and John D Hunter. "Python: an ecosystem for scientific computing". In: *Computing in Science & Engineering* 13.2 (2011), pp. 13–21.

[18] Sergio J. Rey and Elijah Knaap. "The PySAL Ecosystem: Philosophy and Implementation". In: *Center for Geospatial Sciences, University of California, Riverside* (2021). URL: https://cogs.sdsu.edu/publication/rey-2021-py-sal-ecosystem/.

[19] Eduardo Rodrigues, Benjamin Krikler, Chris Burr, Dmitri Smirnov, Hans Dembinski, Henry Schreiner, Jaydeep Nandi, Jim Pivarski, Matthew Feickert, Matthieu Marinangeli, Nick Smith, and Pratyush Das. "The Scikit-HEP Project: Overview and Prospects". In: *EPJ Web of Conferences*. Vol. 245. EDP Sciences, 2020, p. 06028. DOI: 10.1051/epjconf/202024506028. URL: https://doi.org/10.1051/epjconf/202024506028.

[20] Scientific Python Community. *Repo-Review: Repository Style and Compliance Checker for Scientific Python*. https://learn.scientific-python.org/development/guides/repo-review/. Guide and tool for reviewing the style and readiness of GitHub repositories in the scientific Python ecosystem. 2024.

[21] Scientific Python Community. *Scientific Python Ecosystem Coordination Specifications (SPECs)*. https://scientific-python.org/specs/. Recommendations and best practices for projects in the scientific Python ecosystem. 2024.

[22] Scientific Python Team. *Scientific Python Lecture Notes*. URL: https://lectures.scientific-python.org/.

[23] Scientific Python Team. *Scientific Python Project Homepage*. URL: https://scientific-python.org/.

[24] Skipper Seabold and Josef Perktold. "statsmodels: Econometric and statistical modeling with python". In: *9th Python in Science Conference*. 2010.

[25] *Statistical Python Project*. https://statistical-python.org/. GitHub organization: https://github.com/statistical-python. 2025.

[26] Daniel Swain and the pyGAM contributors. *pyGAM: Generalized Additive Models in Python*. https://github.com/dswah/pyGAM. A Python package for fitting generalized additive models (GAMs) with flexible, interpretable smoothing. 2025.

[27] Jonathan Taylor, Trevor Hastie, Gareth James, Robert Tibshirani, and Daniela Witten. *ISLP: Data and Code for 'An Introduction to Statistical Learning with Applications in Python'*. https://github.com/intro-stat-learning/ISLP. ISLP package includes datasets, helper functions, and code for statistical learning labs. 2024.

[28] PySAL Development Team. *PySAL: Python Spatial Analysis Library Meta-Package*. Available at https://github.com/pysal/pysal. 2025. URL: https://github.com/pysal/pysal.

[29] Scikit-HEP Development Team. *Scikit-HEP: Metapackage for Particle Physics Data Analysis*. Available at https://github.com/scikit-hep/scikit-hep. 2025. URL: https://github.com/scikit-hep/scikit-hep.

[30] The pandas development team. *pandas-dev/pandas: Pandas*. Version latest. Feb. 2020. DOI: 10.5281/zenodo.3509134. URL: https://doi.org/10.5281/zenodo.3509134.

[31] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors. "SciPy 1.0—Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17 (2020), pp. 261–272.

[32] Stéfan van der Walt, S Chris Colbert, and Gaël Varoquaux. "The NumPy array: a structure for efficient numerical computation". In: *Computing in Science & Engineering* 13.2 (2011), pp. 22–30. URL: https://doi.org/10.1109/MCSE.2011.37.

[33] Matthew Wardrop and Formulaic contributors. *Formulaic: High-performance Implementation of Wilkinson Formulas for Python*. https://github.com/matthewwardrop/formulaic. A Python package for high-performance dataframe-to-model-matrix conversion and extensible formula parsing. 2024.

[34] Michael Waskom and the seaborn contributors. *seaborn: Statistical Data Visualization in Python*. https://github.com/mwaskom/seaborn. A Python visualization library providing a high-level interface for attractive statistical graphics. 2024.

NSF POSE: Phase II: An Open-Source Ecosystem for Statistical Python

Data Management and Sharing Plan

1. **Types of Data Collected.** In Phase II, the project will support the implementation and growth of the Statistical Python Open Source Ecosystem. Primary data artifacts will include meeting notes, anonymized feedback from users, researchers, students, and partners, design documentation, assessments of governance, technical infrastructure, and software engineering best practices. Additional data may arise from ongoing development and community engagement activities, training events, and roadmap summits. These assessments and materials may inform project reports, published documentation, and future funding proposals.

2. **Dissemination of Data.** All meeting notes, anonymized feedback, technical and governance documentation, assessments, and related materials will be developed in public and remain openly accessible. Outputs will be published on GitHub repositories affiliated with the Statistical Python Project, or hosted on the project's public website.

3. **Licensing of Data.** All data artifacts produced will be published under the CC0 license (`https://creativecommons.org/share-your-work/public-domain/cc0/`), ensuring broad reuse. Feedback and other collected information will not include personally identifiable private data. All participants in meetings, surveys, or community activities will be informed of the data management plan in advance. Any code developed as a side effect of project activities will be released under the MIT or Modified BSD License.

4. **Data Retention.** Collected data is intended to facilitate the growth and sustainability of the Statistical Python ecosystem and to support community transparency. Beyond standard public release on GitHub and the project website, no additional long-term archival is planned.