PSF Scientific Working Group Call for Grant Requests
Development and Maintenance of The Carpentries Python Curriculum

Vinícius W. Salazar
Department of Systems and Computer Engineering
Federal University of Rio de Janeiro
viniws@gmail.com

February 12, 2021

Introduction

The Carpentries is one of the world's leading projects for inclusive data and coding training for scientists. It provides hands-on workshops that are offered by a volunteer community of instructors, with all lessons' content being freely reusable under a Creative Commons Attribution license [1]. From 2012 to 2018, The Carpentries has hosted over 2900 workshops across more than 50 countries, teaching thousands of researchers [2]. In Carpentries workshops, lessons are delivered with evidence-based learning practices [3] that are taught to instructors during their own training. These lessons promote best practices in scientific computing [4, 5], which have been deemed a fundamental, yet often neglected, aspect of modern scientific research [6]. During the year 2020, approximately 200 workshops have been taught globally, despite unprecedented circumstances, as most (if not all) workshops have been shifted to a remote format. The Carpentries enforces a well-defined Code of Conduct, to foster a welcoming and inclusive learning environment [1], and features an extensive assessment program, providing evidence of its impact on learners [7]. This displays the role of The Carpentries in the global scientific community: a project that empowers researchers by teaching them new skills and, while doing so, connects people through education.

This proposal focuses on developing and maintaining The Carpentries Python curriculum. Usually, this task is performed through volunteer efforts across the Carpentries community. However, the lack of dedicated development efforts implies that there is a constant backlog of work to be completed. There are many outstanding needs in the Python curriculum (e.g. lessons to be maintained [8], lessons to be completed, outdated documentation) which could be resolved with such efforts, as they will be detailed throughout this proposal. We request funding for a given number of hours of development work on Python-related lessons in the Carpentries material. Having dedicated developer time for this will substantially improve the quality of this material and, by consequence, contribute to freely-available, high-quality educational resources about the Python language. It is important to note that this proposal is an independent request by a volunteer Carpentries instructor, but it is supported by and in collaboration with The Carpentries as a whole.

In the following sections, we will outline the details of the proposal. Section 1 describes how Python is present in the Carpentries material: what it is used for, which Python packages are employed, and what domain-specific content is covered by the relevant lessons. Section 2 identifies the main priorities for targeted development efforts in the Python curriculum. These include solving open issues, further development of lessons in early stages, and updating documentation. Section 3 states the main goals and milestones of the proposal. Section 4 lays out how the proposal will be executed, including a timeline for the project. Section 5 describes how the project will be assessed at the time of completion, to produce an Evaluation Report to the funding source, and Section 6 characterizes the budget for the project.

1 The Carpentries Python Curriculum

The Carpentries started in 1998 as Software Carpentry, but twenty years later, in 2018, Software Carpentry and its twin program, Data Carpentry, were merged into The Carpentries. In the same year, Library Carpentry was approved as a third lesson program. While the Software Carpentry material is comprised of independent (albeit related) lessons, the Data Carpentry features "curricula", which are sets of domain-specific lessons directed to scientists of that particular area of knowledge [9], e.g. the genomics or the geospatial curriculum. Throughout this document, we are going to refer to the set of Python-related lessons in the Carpentries material as the "Carpentries Python curriculum" (CPC). Although this distinction does not exist in the official material, it will be useful here as these lessons are the focus of this proposal. Currently, there are seven Python lessons in the Carpentries material, with three being stable, and the other four in "alpha" development stage, meaning that they have not been integrated officially in the Carpentries material (Table 1).

Lessons program	**Curriculum**	**Lesson**	**Status**
Software Carpentry	—	Programming with Python	Stable
	—	Plotting and Programming in Python	Stable
Data Carpentry	Ecology	Data Analysis and Visualization in Python for Ecologists	Stable
	Social Science	Data Analysis and Visualization in Python for Social Scientists	Alpha
	Image Processing	Image Processing in Python	Alpha
	Community-contributed	Python for Atmosphere and Ocean Sciences	Alpha
	Astronomy	Python for Astronomy and Physics	Alpha

Table 1: The "Carpentries Python Curriculum" (CPC): lessons in the Carpentries material which use the Python language as a medium.

In addition to those, there are numerous (Python) lessons in "pre-alpha" stage in the Carpentries Incubator [10], which is a collaborative space for lessons contributed by the community. For sake of conciseness, this proposal will focus primarily on the lessons displayed in Table 1, although working in the Carpentries Incubator lessons may be considered should the priority arise.

Lessons in the CPC employ many scientific packages, including, but not limited to Numpy, Scipy, Pandas, Jupyter, Matplotlib, Xarray, and Plotnine [11–17]. They are used extensively for data processing, analysis, and visualization, except for Jupyter, which is used as the interactive environment during the workshops. Taking a Carpentries workshop is the first exposure of many research scientists to these packages, highlighting the necessity of properly maintaining and updating these lectures, as they constitute a valuable resource for scientific Python educators globally.

2 Current Needs

Priority development tasks in the CPC can be divided into three groups:

- Solving open issues in stable lessons;
- Developing lessons in the alpha stage;
- Updating documentation for younger Python versions.

The first group of tasks regards the maintenance of stable lessons. At the time of writing, there are, in total, 37 open issues with a "help wanted" label in the GitHub repository of these lessons and additional ones for lessons in earlier development stages [8]. If we count all open issues, even the ones without the "help wanted" label, there are hundreds more. Closing all "help wanted" issues and mapping the priority of the remaining open issues would be a great enhancement to the CPC as a whole, and alleviate the burden on the volunteer maintenance community.

The second group is focused on continuing the development of lessons that are not yet part of the official Carpentries material ("alpha" status in Table 1). Because these lessons were created by multiple collaborators, this involves contacting these people and discussing with them to identify potential priority points that need to be worked on. If sufficient improvements are made, these lessons may be suitable for integration in the official Carpentries lessons.

The third group of tasks encompasses general maintenance work in the documentation of the lessons. Many lessons were developed before the release of more recent Python versions, (i.e. 3.7, 3.8, and 3.9) and may not be fully updated with modern Python features, such as f-strings, built-in breakpoints, data classes, improved type hints, standard library improvements, and so on. Lessons in the Carpentries material contain "Instructor notes" which are documents to support workshop instructors when teaching. Documentation work would also contemplate updating these notes, and any other relevant supplementary material.

3 Objectives

The reason for this request is to fund dedicated developer hours to work on the three aforementioned groups of tasks. So, for each group, there is a respective objective to be achieved: to close all issues with the "help wanted" label; to improve lectures in the "alpha" stage sufficiently to the point they can be considered for integration in the official material; to update the documentation and supplementary information of all lessons in "stable" stage. We deem this to be feasible given the amount we have requested. If all these objectives are completed before the expected time frame, these objectives can be further extended: to close additional priority issues in each lesson's GitHub repository, to contribute to more lessons in the Carpentries Incubator (which are in "pre-alpha" stage), and to draft a proposal for a new lesson. For this latter objective, there is specific interest for developing a Python lesson for the Genomics curriculum, seeing as there is a wealth of well-established Python libraries for analyzing biological data, such as BioPython [18] and Scikit-Bio [19]; this effort would be secondary, however, to the previously outlined objectives, and would only be considered if the first set of objectives were completed first.

4 Implementation Plan

To tackle the objectives laid out in the previous section, we propose an implementation plan to organize development efforts. It can be summarized as a list of activities with an expected number of hours to be performed (Table 2). The activities sum up to a total of 160 hours of work, to be executed between the months of April and May 2021. If the activities are completed within this expected number of hours and before the expected time frame, the extended objectives described in the previous section may be pursued. The bulk of work hours will be dedicated to development work, but activities for planning, discussion with the community, and evaluation are also contemplated.

5 Evaluation Plan

Following the Python Software Foundation's guidelines for grant requests, a final report will be produced at the end of the proposed time frame, detailing the work which was executed. We allocate specific activities to assess whether the proposal's goals have been met (Table 2). An adequate metric for gauging that is the number of issues closed, assuming that new issues will also be created during the implementation plan. There may be issues, however, where work may be done and the issue remains open. The final report should be able to describe this and identify potential tasks remaining.

Activity	Description	Expected time
Map issues	Revise the list of "help wanted" issues and define the priority status of each issue.	6 hours
Discuss with contributors	Contact contributors of the lessons in "alpha" stage and identify pending tasks in each lesson. Translate these tasks into issues.	4 hours
Review previous community discussions	Review past community discussions that may be relevant to the proposal's objectives. The Carpentries keeps archives of these activities, so they are promptly available.	4 hours
Host community discussion	Host a community discussion on the Carpentries Python Curriculum. Raise suggestions from the community and translate them into development tasks and issues.	10 hours
Close "help wanted" issues	Perform development work on "help wanted" issues and submit the corresponding PRs to the official Carpentries repositories	**60 hours**
Improve lessons in "alpha" stage	Perform development work on lessons in "alpha" stage and close the corresponding issues.	**60 hours**
Mid-proposal review	Halfway through the proposed time frame, conduct a review on the milestones achieved so far, and adjust priorities accordingly.	4 hours
Final assessment	At the end of the proposal's time frame, conduct a review of all milestones achieved and difficulties encountered, and produce a final report for submission to the funding source.	12 hours

Table 2: List of activities in the implementation plan.

6 Budget

We request an amount of $4000 USD to carry out this proposal. This will be fully directed to fund dedicated developer work during the months of April and May 2021, covering 160 hours of work, as detailed in Table 2. Included in the costs are network and power expenses and bank transfer fees. To properly justify the usage of these funds, a final report will be produced as described in Sections 4 and 5. This proposal is flexible and we are open to negotiating any aspects of it.

Concluding remarks

This proposal is a request to fund developer work on The Carpentries Python curriculum. The lessons included in this material cover a range of scientific topics, such as ecology, image processing, and social science. They teach how to use specialized scientific Python packages for domain-specific applications, and serve as a hands-on introduction to scientific Python for many researchers. Thus, providing dedicated maintenance and development work for such educational resources will greatly benefit not only the supporting project, but hundreds of educators and thousands of learners who make use of this material.

References

[1] The Carpentries, "The carpentries handbook," 2021. [Online]. Available: https://docs.carpentries.org/

[2] ——, "Workshops," 2021. [Online]. Available: https://carpentries.org/workshops/

[3] S. A. Ambrose, M. W. Bridges, M. DiPietro, M. C. Lovett, and M. K. Norman, *How learning works: Seven research-based principles for smart teaching.* John Wiley & Sons, 2010.

[4] G. Wilson, D. A. Aruliah, C. T. Brown, N. P. C. Hong, M. Davis, R. T. Guy, S. H. D. Haddock, K. D. Huff, I. M. Mitchell, M. D. Plumbley, and Others, "Best practices for scientific computing," *PLoS Biology*, vol. 12, no. 1, p. e1001745, 2014.

[5] G. Wilson, J. Bryan, K. Cranston, J. Kitzes, L. Nederbragt, and T. K. Teal, "Good enough practices in scientific computing," *PLOS Computational Biology*, vol. 13, no. 6, p. e1005510, jun 2017. [Online]. Available: https://dx.plos.org/10.1371/journal.pcbi.1005510

[6] M. Ponce, E. Spence, R. van Zon, and D. Gruner, "Bridging the Educational Gap between Emerging and Established Scientific Computing Disciplines," *The Journal of Computational Science Education*, vol. 10, no. 1, pp. 4–11, 2019.

[7] The Carpentries, "Assessments and impact," 2021. [Online]. Available: https://carpentries.org/assessment/

[8] ——, "List of GitHub help wanted issues," 2021. [Online]. Available: https://carpentries.org/help-wanted-issues/

[9] ——, "Data carpentry lessons," 2021. [Online]. Available: https://datacarpentry.org/lessons/

[10] ——, "The carpentries incubator," 2021. [Online]. Available: https://github.com/carpentries-incubator/proposals/#the-carpentries-incubator

[11] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith *et al.*, "Array programming with numpy," *Nature*, vol. 585, no. 7825, pp. 357–362, 2020.

[12] P. Virtanen, R. Gommers, T. E. Oliphant *et al.*, "SciPy 1.0: fundamental algorithms for scientific computing in Python," *Nature Methods*, vol. 17, no. 3, pp. 261–272, mar 2020. [Online]. Available: http://www.nature.com/articles/s41592-019-0686-2

[13] W. McKinney, "pandas: a foundational Python library for data analysis and statistics," *Python for High Performance and Scientific Computing*, vol. 14, no. 9, 2011. [Online]. Available: https://www.dlr.de/sc/portaldata/15/resources/dokumente/pyhpc2011/submissions/pyhpc2011{_}submission{_}9.pdf

[14] M. Ragan-Kelley, F. Perez, B. Granger, T. Kluyver, P. Ivanov, J. Frederic, and M. Bussonnier, "The Jupyter/IPython architecture: a unified view of computational research, from interactive exploration to communication and publication." in *AGU Fall Meeting Abstracts*, 2014.

[15] J. D. Hunter, "Matplotlib: A 2D Graphics Environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007. [Online]. Available: http://ieeexplore.ieee.org/document/4160265/

[16] S. Hoyer and J. Hamman, "xarray: Nd labeled arrays and datasets in python," *Journal of Open Research Software*, vol. 5, no. 1, 2017.

[17] Hassan Kibirige, "Plotnine: a grammar of graphics for python," 2021. [Online]. Available: https://github.com/has2k1/plotnine

[18] P. J. A. Cock, T. Antao, J. T. Chang, B. A. Chapman, C. J. Cox, A. Dalke, I. Friedberg, T. Hamelryck, F. Kauff, B. Wilczynski, and M. J. L. de Hoon, "Biopython: freely available Python tools for computational molecular biology and bioinformatics," *Bioinformatics*, vol. 25, no. 11, pp. 1422–1423, jun 2009. [Online]. Available: https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btp163

[19] Scikit-bio, "Scikit-bio," 2021. [Online]. Available: http://scikit-bio.org/