TLEF Project – Final Report

Report Completion Date: (2023/05/31)

1. PROJECT OVERVIEW

1.1. General Information

Project Title:	Embedding Open source Computational Tools into the Quantitative Earth Science Specializations		
Principal Investigator:	Tara Ivanochko		
Report Submitted By:			
Project Initiation Date:	2020/05/01	**Project Completion Date:**	2023/04/30
Project Type:	☒ Large Transformation ☐ Small Innovation ☐ UDL Fellows Program ☐ Hybrid and Multi-access Course Redesign Project ☐ Other: [please specify]		

1.2. Project Focus Areas – *Please select all the areas that describe your project.*

☒Resource development (e.g., learning materials, media)

☒ Infrastructure development (e.g., management tools, repositories, learning spaces)

☒ Pedagogies for student learning and/or engagement (e.g., active learning)

☐ Innovative assessments (e.g., two-stage exams, student peer-assessment)

☒ Teaching roles and training (e.g., teaching practice development, TA roles)

☐ Curriculum (e.g., program development/implementation, learning communities)

☐ Student experience outside the classroom (e.g., wellbeing, social inclusion)

☐ Experiential and work-integrated learning (e.g., co-op, community service learning)

☐ Indigenous-focused curricula and ways of knowing

☐ Diversity and inclusion in teaching and learning contexts

☒ Open educational resources

☐ Other: [please specify]

1.3. Final Project Summary – *What did you do/change with this project? Explain how the project contributed toward the enhancement of teaching and learning for UBC students.*

What was done or changed:

- **Python** has been introduced for five courses, transforming all learning materials and activities from R or Matlab. **Jupyter Notebook** (JNB) environments are used in all cases.
- Fifteen **dashboard apps** were designed & built (many by students) using Python and a consistent interactive library (Plotly) to enable interactive exploration of concepts or datasets. Corresponding pedagogy was established for optimizing students' learning. Nine courses currently use these, and others have expressed interest.
- **Guidelines** and other resources written as open source "executable books" are being produced regarding design, construction and use of interactive resources to optimize student learning. These will help instructors, students and administrators use, maintain, and improve these materials and learning activities have been (or are being) generated. See https://eoas-ubc.github.io/. Many will become Open Education Resources within a few months of the OCESE project official close.
- Options for **infrastructure** necessary for delivering dashboards and Jupyter notebooks were explored and chosen. Some courses have students load and maintain Python and JNBs on their own laptops; some courses use UBC's new (and still evolving) open Jupyter hub, and some courses are experimenting with third party hubs via the https://2i2c.org/ organization. Dashboards are provided on a custom server managed by the Department.
- **Consulting and support** for fourteen courses was enabled by OCESE project funding supporting an experienced geoscience education specialist. This has helped instructors and teaching assistants adapt their courses and pedagogic practices to optimize the use of new resources, dashboards and JNB's.
- **For any course,** techniques for automating the management or delivery of *questions, question sets,* and *automated grading* practices has been supported, including code to interface between questions written in Markdown and the Canvas API.
- **Community of practice:** Partly in response to significant challenges experienced in both the first and second offerings of EOSC 211 using Jupyter Notebooks, a new centrally coordinated interdisciplinary committee / community was created to discuss CTLT's open.jupyter.ubc.ca facility. The OCESE project team advocated for this community-of-practice to improve communication among UBC colleagues who need stable, reliable Jupyter Hubs for their courses. The group now meets semi-regularly to discuss campus-wide needs and approaches to supporting open source pedagogies at UBC.

How the project contributed to enhancing teaching

- **Computing & documentation resources, consulting and support** (listed above) help enhance teaching by ensuring instructors and teaching assistants can deliver courses using Python, JNBs, or dashboards.
- **Use of open source materials and practices** has informed instructors of techniques, helped graduate and undergrad students who worked on the project develop career-relevant teaching and technical skills, and moved teaching resources (e.g. static or interactive textbooks, and others) from private sources to more modern and versatile open source environments.

- Establishing the necessary **infrastructure and clarifying responsibilities** for maintaining these technical cloud computing facilities has been challenging but is critical for sustaining the use of these teaching and learning resources.
- Teachers are more likely to rely on **students' prior knowledge** if earlier courses are taught using the same programming language and analysis techniques that will be used in senior courses.
- **Transferring teaching tactics** to instructors who are new to a course can be challenging. OCESE project documentation and professional development products aim to ease that transition, and help instructors and teaching assistants efficiently adopt the teaching tactics and resources developed during the project.
- Based to some extent on experiences in OCESE, interest in employing dashboards and/or Jupyter notebooks in EOAS courses has grown substantially. Based on polling within the department, of 78 courses within the department, 32 now either use or hope to use open source computing resources.

How the project contributed to enhancing learning

- Two major development choices were to establish Python as the programming language used in all ATSC, ENVR and EOSC courses that involve computing, and to deliver that education using Juptyer notebooks. These choices contribute to enhanced learning because Python has become the most popular programming language for computing in the environmental, atmospheric and Earth sciences. It is also a principal teaching language for UBC computer science and data science coursework. Jupyter Notebooks are efficient, versatile and sustainable as they have become a standard platform for teaching and learning computing and data science across disciplines. These two choices are also consistent with practices in the disciplines and across UBC, and they deliver education that is relevant for any occupation that students will pursue after graduating with an EOAS degree.
- The OCESE project partnered with climate-science education development projects, leveraging the ideas, funding and personnel to benefit student learning in EOSC 112, EOSC 340 and EOSC 442.
- Students can tackle more advanced material in senior courses if their prior learning is designed to be consistent throughout their degree program. Benefits of having students learn and use Python across the EOAS curriculum are expected to emerge in 2-4 years following the conversion of EOSC 211 from MatLab to Python in the fall term of 2021.
- Exposing students to open source materials and practices for programming or analyzing and interpreting data provides them with skills and confidence that can be applied far beyond the confines of individual courses or learning situations. These are highly portable capabilities that will benefit students in whatever pathways they pursue during their schooling and beyond.
- Feedback from students in EOSC 112, 325, 372 and ENVR 300 indicates that exploring data and quantitative concepts or solving meaningful problems with dashboards was inspiring, "fun", and helped relate new ideas to real world applications. This feedback is consistent with precedent & literature regarding the use of interactive simulations when they are carefully integrated into learning activities and assignments.
- We have learned useful lessons about how to ensure students get as much as possible out of learning with interactive resources, while keeping the time and effort of instructors to a minimum. See for example "Developing dashboards" or project results for specific courses under "Course enhancements".

- Time and funding from the OCESE project have enabled the Statistics department to offer a Python-based section of DSCI 100. Student learning is enhanced by providing the option to learn at this early stage in either "R" or Python.
- Open source infrastructure & procedures developed or enhanced during OCESE are enabling senior undergraduate students to pursue challenging and meaningful projects involving sophisticated computing and large data sets in senior courses and in directed studies projects.
- Although not a part of the original project proposal, aspects of inclusion and accessibility related to learning data science have been considered during the OCESE project and communicated within UBC. See for example a discussion panel on this topic at UBC Celebrate Learning week, May 13, 2022. In short, the need for stable, reliable cloud-hosted hubs supplied by the institution is critical for ensuring students can learn with dashboards and Jupyter notebooks even if their personal computing capacity is limited to low-cost laptops, Chromebooks or tablets.

NOTE: For an executive summary of project outcomes, please see https://eoas-ubc.github.io/execsum.html.

1.4. Team Members – *Tables of faculty and students who participated in the project.*

Team members who contributed as instructors

Name	Title/Affiliation	Responsibilities/Roles
Tara Ivanochko	Teaching Prof., Fac. Sci., Dep't EOAS	Project P.I., instructor ENVR 300
Susan Allen	Prof., Fac. Sci., Dep't EOAS	Instructor, EOSC 471
Phil Austin	Assoc. Prof., Fac. Sci., Dep't EOAS	Principle project developer, instructor EOSC 340, ATSC 301
Roger Beckie	Prof., Fac. Sci., Dep't EOAS	Instructor EOSC 429
Michael Bostock	Prof., Fac. Sci., Dep't EOAS	Instructor EOSC 354
Catherine Johnson	Prof., Fac. Sci., Dep't EOAS	Instructor EOSC 211
Valentina Radic	Assoc. Prof., Fac. Sci., Dep't EOAS	Instructor ENVR 300
Stephanie Waterman	Assist. Prof., Fac. Sci., Dep't EOAS	Instructor EOSC 112, 372
Lindsay Heagy	Assist. Prof., Fac. Sci., Dep't EOAS	Instructor EOSC 350, DSCI 100, and open source computing expertise.
Kristin Orians	Assoc. Prof., Fac. Sci., Dep't EOAS	Instructor EOSC 372
Maite Maldonado	Prof., Fac. Sci., Dep't EOAS	Instructor EOSC 372
Stuart Sutherland	Teaching Prof., Fac. Sci., Dep't EOAS	Instructor EOSC 116, 425
Ali Ameli	Assist. Prof., Fac. Sci., Dep't EOAS	Instructor EOSC 325
Rachel White	Assist. Prof., Fac. Sci., Dep't EOAS	Instructor EOSC 340
Anais Orsi	Assist. Prof., Fac. Sci., Dep't EOAS	Instructor EOSC 112
Lucy Porritt	Lecturer, Fac. Sci., Dep't EOAS	Instructor EOSC 323
Louise Longridge	Lecturer, Fac. Sci., Dep't EOAS	Instructor EOSC 116, 326
Reid Merrill	PhD student	Instructor EOSC 354
Joseph Capriottie	PostDoc	Instructor EOSC 350
Sam Anderson	PhD student	Instructor EOSC 410
Michael Lipsen	Lecturer, Fac. Sci., Dep't EOAS	Instructor EOSC 422
Ben O'Connor	PhD student	Taught EOSC 422 computing labs

Student contributors: Most were hired with support from the UBC WorkLearn program:

Name	**Role**	**Responsibilities / duties**
Carol Zhang	undergrad	programming question sets & Canvas interface
Jamie Byer	undergrad	Supported Python conversion of EOSC 354, & programming dashboards
Danil Platonov	undergrad	programming early versions of server software
Benjamin Chang	undergrad	programming related to question sets & interacting with the Canvas API.
Mara Colclough	undergrad	programming early versions of python resources
Iddo Sadeh	undergrad	Python conversion of EOSC 442, 471, DSCI 100 & dashboard programming
Navya Dahiya	grad	supported DSCI 100 Python conversion
Wanying Ye	grad	supported DSCI 100 Python conversion
Andrew Loeppky	grad	supported EOSC 211 Python conversion & implementation
Hariharan Umashankar	grad	various programming, including dashboards
Jacob McFarlane	grad	work on advanced dashboard programming for EOSC 340
Yiki Su	grad	supported DSCI 100 Python conversion
Chris Rodell	grad	volunteer - supported conversion of EOSC 410
Francis Rossman	grad	supported EOSC 211 conversion, and dashboard programming

1.5. Courses Reached – *Courses reached by the project, including courses not mentioned in the original proposal (adapt to the context of your project if necessary).* Courses affected during the OCESE project:

Courses involved, enrolments & years affected.

Courses	No. students impacted by OCESE			
	2020	2021	2022	subtotals
ATSC 301			21	21
DSCI 100			68	68
ENVR 300		58	63	121
ENVR 420**				0
EOSC 112			212	212
EOSC 116			708	708
EOSC 211		95	109	204
EOSC 310			223	223
EOSC 323^			45	45
EOSC 325		44	75	119
EOSC 326^		183	140	323
EOSC 329**				0
EOSC 340			215	215
EOSC 350	40	43	36	119
EOSC 354	13	10	8	31
EOSC 372		134	157	291
EOSC 373**				0
EOSC 410/510	35	34	48	117
EOSC 425^			9	9
EOSC 429*				0
EOSC 442			56	56
EOSC 471*				0
			TOTAL	**2882**

16 courses in the proposal and affected.

** 2 courses in proposal but affects delayed to 2023W.*

*** 3 courses in proposal but not affected.*

^ 3 courses affected but not in proposal.

Enrolments from *https://pair.ubc.ca/*

2. OUTPUTS AND/OR PRODUCTS

2.1. ***List*** *of project outputs including URLs.*

Output(s)/Product(s):	URL (if applicable):
Fifteen individual interactive dashboards for use in any course that involves learning about corresponding concepts. As of April 2023, dashboards have been used in at least nine different courses.	https://eoas-ubc.github.io/dashboards.html lists all dashboards including URLs to active sites plus URLS to corresponding github repositories.
EOSC 211: All materials, worksheets, assignments and labs converted to Python and Jupyter Notebooks.	https://eoas-ubc.github.io/crs-eosc211.html
EOSC 354: All labs converted to Python and Jupyter Notebooks, including a prior-knowledge self-test.	https://eoas-ubc.github.io/crs-eosc354.html
EOSC 471: All labs converted to Python and Jupyter Notebooks. To be piloted in fall, 2023.	https://eoas-ubc.github.io/crs-eosc471.html
DSCI 100's new Python-based section, including all labs, worksheets, a new textbook and corresponding teaching materials. DSCI 100 is a statistics course and Stats faculty are in charge of running and maintaining the course. The OCESE project's contributions included support for students working on materials, textbook, assignments, etc.	- https://eoas-ubc.github.io/crs-dsci100.html - https://github.com/UBC-DSCI/dsci-100-student-python - https://python.datasciencebook.ca/index.html
Complete project documentation is available as an opensource Jupyter Book. It includes details about impacts for **every course involved**, progress reports, and **15 tutorials or "how-to" guides** for using open source resources and practices that were implemented during the OCESE project.	https://eoas-ubc.github.io/index.html
OCESE project provided **consulting** for all instructors and teaching assistants involved in the 22 courses affected by OCESE between 2020 and 2023.	Details for each course begin at https://eoas-ubc.github.io/course_materials.html
Detailed wisdom was gained by OCESE project lead P. Austin regarding use of Jupyter hubs for teaching at all levels, deploying dashboards on custom servers, and tactics for including automatic grading. These and other technical details are critical for efficient, effective use in teaching, and for sustaining these teaching and learning capabilities into the future.	Transferring detailed new technical knowledge about how open source resources and practices can be effectively and sustainably deployed are being prepared, and will be visible as they are produced at the OCESE project website: https://eoas-ubc.github.io
Automatic grading strategies and managing question sets (e.g. interfacing with Canvas via its Application Programmer's Interface or API) was explored in the context of EOSC211, EOSC340, and EOSC325. Techniques and tools evolved rapidly between 2020 and 2023, both at UBC and in the open source community. Experiences and recommendations are documented at the URL.	https://eoas-ubc.github.io/openassessment.html

2.2. Item(s) Not Met – *List of intended outputs that were not completed and the reason(s) for this.*

Item(s) Not Met:	Reason:
The main goal that was not met was delivery of workshops on best practices	Two main reasons: 1) COVID pandemic began at the same time as the OCESE project started, resulting in instructors who were too busy adapting to online learning to consider the learning

for using open source resources and techniques, including Python, cloud computing (hubs), dashboards, GitHub, Jupyter books, auto-grading, and others.	planned for OCESE project components. Consequently, faculty professional development and training became mainly one-on-one support for instructors and teaching assistants of specific courses participating in OCESE. 2) The proposal was overly ambitious regarding readiness to deliver training. Much was learned about use of open source resources and techniques, teaching with python and Jupyter notebooks, etc. but those lessons were learned during the project. Even during the last few months, project personnel were still refining tactics for delivering education with these materials and methods. Training faculty can begin only after OCESE deliverables are considered "stable" for use in courses.
ENVR 420: Convert "R" exercises to Python.	This course was changed to a summer course, then back to normal. The instructor was not prepared to add OCESE adjustments to these changes during the time available.
EOSC 329: Dashboard design and use was anticipated but not accomplished.	This course was taught by a different instructor every year (including sessionals) during the OCESE project. Therefore, changes were considered inadvisable during these three years. Dashboards produced for EOSC 325 (a similar course in hydrogeology) could be incorporated into EOSC 329 if future instructors desire.
DSCI 100: Generate a course section to be taught in Python USING EARTH SCIENCE CONTEXTS	This section was successfully piloted in term 2, 2023W. The team chose to teach in Python, but using the same data sets as the R-version. Adapting these contexts to make use of Earth science data sets will be carried out for a future teaching session.

3. PROJECT IMPACT

3.1. Project Impact Areas – *Please select all the areas where your project made an impact.*

☒ Student learning and knowledge
☐ Student engagement and attitudes
☐ Instructional team-satisfaction
☒ Teaching practices
☐ Student wellbeing, social inclusion
☐ Awareness and capacity around strategic areas (Indigenous, equity and diversity)
☒ Unit operations and processes
☐ Other: [please specify]

3.2. Details on each of the impact areas you selected in 3.1

3.2.1. Student learning and knowledge

- Students taking **EOAS electives at all levels** (e.g. EOSC 112, 211, 325, 340, 429, 442 and others) now explore concepts or data sets with focused interactive dashboards and/or prepared Jupyter Notebooks. Topics include oceanography, basic climate science, climate data sets and models, several concepts in hydrogeology, geological structures and applied & planetary geophysics. Use of interactive dashboards helps students focus on important concepts without being constrained by having to fetch, wrangle, analyze or otherwise manipulate complex data sets or equations.
- More than half the dashboards are directly related to **climate concepts,** consistent with the increased interest among students in learning more about climate science.
- Students in EOAS degree programs now **learn to program in Python early in their degree** (e.g. in EOSC 211 and DSCI 100), and then further develop those skills in subsequent courses in geophysics, atmospheric sciences and oceanography (e.g. EOSC 354, 410, ATSC 301 and others).

- Students in some core 3rd and 4th year EOAS courses (e.g. ATSC 301, 409, EOSC 410) gain experiences related to using **open source programing practices** (eg GitHub version control and collaboration) and open datasets from satellites, climate models (e.g. the CMIP6 archive), oceanographic data sets and geophysical forward, inverse and machine learning modeling procedures. Having a consistent programming environment across the department means students can progress more quickly and gain more advanced capabilities than before, when courses used a wider variety of settings.
- Learning **Python instead of focusing on MatLab** as an environment ensures students graduate with immediately applicable skills that rely on dependable open source resources and practices rather than being constrained by having learned with more restricted and expensive commercial facilities.
- **Effectiveness of Python** with learning tasks delivered as Jupyter Notebooks: When the TA supporting EOSC442 was asked: *Is the Jupyter notebook environment working for them (and you?),* his written response was:

  > *From a programming perspective, it is not my favorite working environment, although I admittedly have not taken the time to learn the "ins and outs" of JNB and customize it to my preferences. That's possibly just my grumpy MATLAB-user voice speaking.*
  >
  > *From a teaching/learning perspective however, I think it is really quite good. A large majority of the students that attend the labs are completing the assignments (more or less) within the allotted lab time. This was far from the case last year and was a major goal in the assignment conversion/updates. Part of this is because the students seem more generally engaged this term, but a large part of it is (I think) due to* ***the increased number of examples, hints, and general "hand-holding" that can now be contained within the notebooks****.*

  - **Comment from EOSC 442 TA**: *"The course is recognized as way too much work for a single credit, but this is well known and being considered at the curriculum level. That said, I do think that the course is appreciated by students that genuinely care about developing their programming skills. "I learned SO much from Lab 1A" was something that I overheard as a group of students were leaving class several weeks ago. Even though the workload/credit balance is off, I think students do feel like they walk away with a usefully-expanded skill set. I think it helps take them from 'kind of knowing Python' to feeling more confident that they can put it on their resume as a skill."*

3.2.2. Teaching practices: changes and challenges

Changes are challenging when everyone is "changing" rapidly to teach online. In other words, planned adjustments, whether full-scale transformations or less ambitious enhancements were difficult to justify and accomplish during COVID.

- **Executable books** were attempted but found awkward to maintain and sustain given "special" skills necessary. Working within Canvas is an environment reliably known by all UBC instructors. Adding a new "learning management system" requires development of "redundant" skills.
- It was thought that the constant turnover of graduate student **teaching assistants** might present a challenge but in fact they tend to be more "current" than faculty, and they also learn very quickly. However, it is important to ensure teaching assistants are fully prepared for the specifics of each course as they are often more "hands on" with individual students than instructors.

- **Automating assessments**: initial reliance on nbgrader proved technically challenging and resulted in delayed release of grading in year 1 of teaching the updated EOSC211. Custom procedures were developed for year 2. DSCI100 continues to use it but it only works because of special skills of the teaching team (Timbers, Campbell, Heagy and others). Since the start of OCESE, the open source "PrairieLearn" project has become attractive for managing questions, question sets, and randomizing question collections and other uses.
- **Managing questions**: interfacing external question sets with Canvas or managing Canvas question banks from outside Canvas was explored in the first half of the OCESE project. This became less of a priority during the second half of the project, partly due to the adoption of PrairieLearn, mentioned above.
- **Faculty pro-D**: became attractive only within the last 6 months, due to: over-extended faculty between spring 202 and 2023; lack of awareness of possibilities (poor "marketing" on our part); concerns about "costs" in time and energy to change a whole teaching environment; lack of willingness to be "a guinea pig" and put up with inevitable challenges of introducing major changes in courses full of students. One **recommendation** resulting from these experiences is that regular opportunities are needed to inform faculty and demonstrate opportunities and potential of new ideas.
- Some instructors needed to develop **Python programming skills**. Others, especially newer faculty, are already fluent in Python and several are familiar with use of Jupyter notebooks for deploying learning resources and activities.
- **Keeping up to date** in the presence of rapidly evolving computing environments means courses can not remain static in terms of resources, programing techniques, the libraries available for numerical work, data manipulation and visualization, and so on. "Keeping up" is also challenging from a researcher perspective, so it does help that teaching and learning are carried out in the same ways as conducting research.
- Incorporating a new open source resource or dashboard should be similar to introducing any new activity or assignment, or adjusting the way a specific topic is covered. However, this is true only if the new resource is ready and available. **Developing, building and testing new interactive resources** can be more challenging than simply updating content. The conceptual design, building, testing, piloting and finalizing cycle requires both a clear initial vision of goals, and sufficient time and commitment to see the process through at least a couple of teaching terms. Senior undergraduate or graduate students were found to be ideal contributors to these steps, although some degree of coordination between builders and teachers is also needed to ensure success.
- Committing to open source resources or practices requires **commitment** to participating, or at least monitoring, the open source communities. Not all instructors are willing.

3.2.3. Unit operations and processes: changes and challenges

- Design, build, deploy cycle (above)
- Transferring tactics and resources from instructors who contributed to development and implementation to instructors new to the course. This is a "standard" challenge in all courses, but if novel techniques, resources or pedagogies are involved, the sustainability of corresponding courses requires more explicit transfer tactics. Paired teaching is one of the better approaches.

- Curricular commitment to ensure students can benefit from learning sequences throughout their degree programs. Example of a challenge: 213 not used later. Example of benefit: EOSC 354 will be able to save a couple of weeks on basics, thus advancing students' knowledge about time series analysis more efficiently.
- Servers for dashboards
- Hubs for Jupyter notebooks – basic courses and advanced needs. Engagement with the JNB community at UBC and beyond. Benefits of new faculty with knowledge of open source and JNBs
- Collaboration between EOAS and STATs w.r.t. DSCI 100
- Contribution of STLF – needs for internal "marketing" and communication, although COVID did make this difficult.

3.3. How do you know that the impacts listed in 3.1/3.2 occurred? – *Describe how you evaluated changes/impacts (e.g., collected survey data, conducted focus groups/interviews, learning analytics, etc.) and what was learned about your project from the evaluation. You are encouraged to include graphical representations of data and/or scenarios or quotes to represent and illustrate key themes.*

3.3.1. How evaluated

One general measure of success is the rate of adoption of open source practices introduced during the OCESE project across the Department. Most instructors teaching courses that involve computing have committed to teach with Python so students learn quantitative Earth sciences using one language throughout their time in EOAS. Furthermore, there has been wide adoption of Jupyter notebooks as the delivery medium for courses with programing goals, as well as quantitative goals not involving programing. In particular, the newest faculty virtually all use Python in their research, as do nearly all graduate students.

More specific evaluation of OCESE project outcomes depends on the type of change or "product" and the context of their implementation. **Table 5** summarizes the evaluation data obtained across the project, organized by the courses for which data were obtained. Details of evaluation data obtained for each course, including results, are given for each course, starting at https://eoas-ubc.github.io/course_materials.html.

Types of data summarized in Table 5 (columns) include: (a) students' work, or surveys conducted before, during or after the course; (b) saved communication channels used by developers, TAs and instructors (usually email or Slack); (c) feedback from students who worked for the project; (d) analyzed Piazza discussion-board data; and (e) in-person or online class observations.

A rough idea of the extent to which initiatives were implemented for each course is summarized in the last column. Of the 17 courses listed, 9 have committed to permanent adoption of OCESE project contributions, 6 have incorporated the changes but continued use depends on future instructors keeping those changes in place, and 2 courses (both of which were not part of the original proposal) experimented with use of new dashboards at least once but confirmation of continued use has not yet been obtained.

Table 5. Project evaluation data

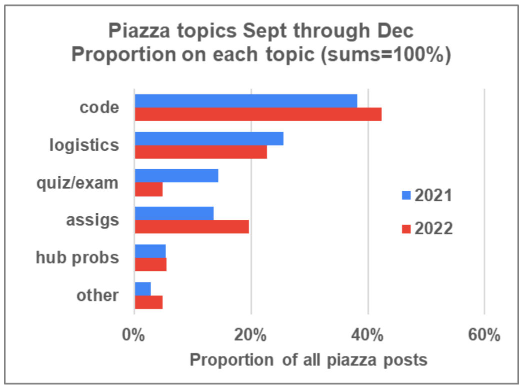
Course	Student work or surveys	Slack, email	student worker feedback	Piazza data	class obs'ns	Adopted*
ATSC 301	2					y
DSCI 100		1	1			y
ENVR 300	2					p
EOSC 112	2				1	p
EOSC 116		1				p
EOSC 116, 326		1				p
EOSC 211	7	2	2	1	1	y
EOSC 310		1				u
EOSC 323		1				u
EOSC 325	3	1			1	y
EOSC 340						p
EOSC 350	1					y
EOSC 354	1		1			y
EOSC 372	2					y
EOSC 410/5	1					y
EOSC 442	1	1	1			y
EOSC 471		1	1			p

**y = Yes, commitment to permanent change*
**p = Probably; yes for some course instructors.*
**u = Unclear; more than one use not confirmed.*

Examples of data gathered:

- **Student work**: In one example, submitted worksheets and student feedback were gathered from a group activity ENVR 300 involving the first iteration of the Atmospheric CO2 dashboard. Results were instrumental in generating a second more capable version. "advanced" projects possible (from TLEF poster #2). See details in OCESE documentation on the summary page for ENVR 300.
- **Student surveys**: Surveys in 10 courses (Table 5 above) provided insights, feedback and quotes from students before, during and at the end of the courses. Data are presented and discussed in OCESE documentation summary pages for each course; see these pages starting here.
- **Slack, email and informal discussion data** were saved to provide anecdotal information about how the development process evolved. This was particularly useful for the most challenging course transformation, EOSC211, during which the instructor was both brushing up on Python programming skills and supervising student workers who were translating (or developing new) learning worksheets and assignments from MatLab to Python.
- **Student worker and TA feedback** including quotes from worklearn project reports related to work for EOSC 211, 354, 442, 471, and dashboards more generally.

- **Piazza discussion board topics** from the first and second year after transforming eosc211 are illustrated in the figure here.
- **Class observations** from EOSC 112 to advise on a new in-class activity that uses IPCC Climate Atlas, 211 (early classes observed using COPUS, and the Python installation day to capture successes and challenges (it went unexpectedly well), 325 to observe the first use of new dashboards.



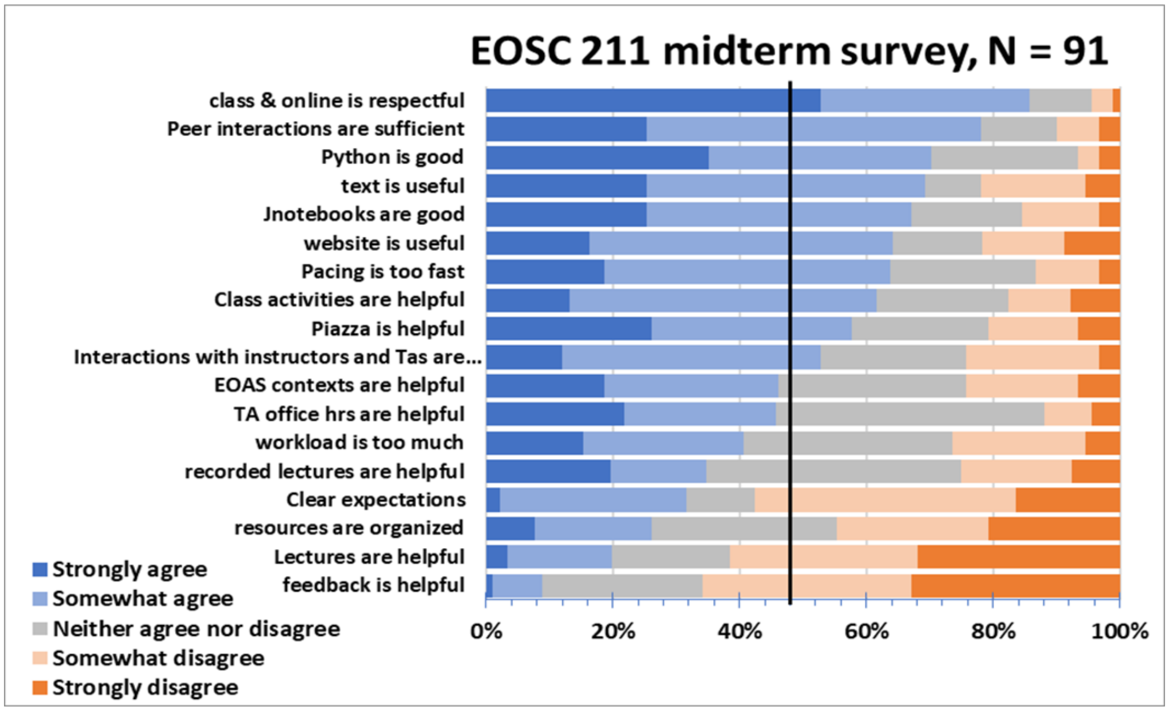
3.3.2. What was learned from evaluation

Comments below are representative of some of the lessons learned during this project, partly from specific evaluation work, but also from experiences carrying out project tasks. More details about "lessons learned" at various stages during the project can be found in presentations and reports; see Progress Reports, as well as poster presentations from May 2023, 2022, 2021, and Dec 2021.

From student work and survey-based feedback: Most students appreciate opportunities to interact with concepts and datasets. Here is a sampling of open-ended feedback:

- EOSC 112 dashboard feedback results:
  - *It explains the effects of the contributing factors in a visual way and was straight to the point when explaining with words.*
  - *Very organized and easy to use, simple and concise explanations and good visuals.*
  - *Suggestion: Maybe add more descriptions on what each contributing factor means and what does the sum selected factors actually do, especially for layperson*
- EOSC 372 student feedback: *I liked how the dashboard exercise applied various concepts using real data from the world's oceans. The online dashboard was quite easy to use and interesting as it allowed us to compare different nutrients/properties profiles in different oceans, synthesizing all the concepts in the course.*
- EOSC 211 student feedback (analyzed & selected):
  - What was helpful? *"worksheets", "practice problems", "TAs are great", "peers"*.
  - What was challenging? *"organization", "coding and understanding the question", "the labs"*.
  - Recommendations: *"more feedback", "explain code line-by-line", "focus in lectures"*.

The image below provides one example of feedback data gathered in a course gathered midway through the first term of changes made in EOSC 211, "*Computer Methods in Earth, Ocean and Atmospheric Sciences*".



Student survey feedback (as well as anecdotal in-class experiences) also highlight the importance of stabilizing the delivery of Jupyter hubs and other server-based resources, whether “in-house” or on cloud-based computing platforms. Details about survey data and implications can be found on the OCESE documentation pages for each course.

A few quotes from instructors:

- EOSC 372 Instructor feedback: “*I am so impressed ... I love how (a) sliders constrain and adjust axes, (b) data at various real stations can be chosen on a map and compared, (c) results can be saved to submit for assessment. I agree that now is the time to finalize an assignment, so thank you to the team!*”
- From ENVR 300, paraphrased: “*The in-class group activity that used the atmospheric CO2 dashboard worked very well. I will likely use the same procedure when we are in-person again.*”
- From E325; email end of term: “*Thanks so much for detailed assessment. Students' suggestions are very helpful and I will reflect on them for the rest of this term and certainly incorporate them in the course design next fall.*”

From slack, email and informal discussions with teaching teams: these collections of communications were instrumental in keeping abreast of progress, challenges and solutions for the variety of project threads. Lessons learned from these sources are primarily about coordination and execution of educational development projects.

For example, support from graduate students who are simultaneously assigned as teaching assistants has been critical for enabling rapid adjustments when problems occur or providing timely points-of-contact for students. They also can carry out the work of writing, coding, testing and assessing that instructors may have little time for, during development and piloting any major or minor adjustments to a course.

Another lesson learned from engaging with instructors is that some need support to conceive, design and deploy the active learning opportunities enabled by dashboards or Jupyter notebooks. Of course, some instructors are already highly "action oriented" in their pedagogy. However, in other cases it was worth spending OCESE project time helping instructors adjust their teaching strategies so that students can be more engaged with concepts and data sets. This requires repeated engagement at all stages of course design, modification and delivery, as well as followup after teaching the section, and in subsequent terms. These innovations then have to survive future transfers of the course to instructors who were not involved in development, and this is not always guaranteed.

Regarding course conversions to Python (from any other language), the OCESE project has made it clear that converting beginner's courses are challenging & costly because "everything" must be changed. In comparison, converting courses that expect students to have had at least some prior computing experience is much more straightforward, especially if converting assignments or labs only. However, for any such conversion, the choice of how Jupyter Notebooks or dashboards are served must be made carefully because these hubs or servers must be "bomb-proof" and scalable. Otherwise student learning may be compromised and instructors will experience a highly stressful term.

From student workers and TA feedback: student contributors, whether employed as project workers (usually with UBC Work Learn support) or as teaching assistants, have been a lynch-pin for this project. They carried out all dashboard programming and much of the heavy-lifting of translating courses to Python. All these contributors provided important, insightful and creative input that ensured implementation of most initiatives were as trouble free as possible. Nearly all provided spontaneous, or requested reflections on their work which recognized how much they learned about teaching, learning, coding, or project management. Two workers also sent word from their subsequent employment (after graduating) expressing satisfaction with how lessons they learned as members of this team contributed directly to success in their workplaces.

E.g.: *"my boss has asked me to convert all our data quality control from excel to python, and to bring the whole team up to speed on doing analysis with Python, which basically means delivering a highly condensed version of EOSC211 to our atmospheric team. … I am using pretty much all the skills I developed over the last 2 years!"*.

Other quotes and reflections are being gathered for the OCESE project documentation page about training of faculty and students.

From Piazza discussion board topics: Student concerns raised on the Piazza platform seem to be relatively similar in both terms, except for posts regarding quizzes and exams. The higher proportion of these posts in 2021 relates to difficulties with the autograding software. In general, it seems that logistical or infrastructure challenges were less concerning to students than they were to instructors. Students are forgiving and adaptable but are most concerned when grading and turn-around of grades is compromised. More details are given on the OCESE documentation page for EOSC 211.

From classroom observations: Courses observed are well taught, perhaps not unexpectedly, since faculty keen to participate in education development projects are already implementing "best practices''. Widespread use of open source resources and tactics is still evolving, partly because the open source communities are still fine-tuning their materials, even while testing them in "live" courses. Observations have also shown that some aspects we worried about are less troublesome than anticipated. For example, setting up consistent installations of Python and Jupyter notebook software on all students' laptops was expected to be labour intensive. But this challenge can be mitigated by careful preparation of "installation environments'' followed by one 45 minute lesson (with extra TA support) devoted to installations. For details see the OCESE documentation Jupyter Notebook Startup page.

Observing initial use of dashboards in lessons enabled developers to offer advice on refining the workflow during the lesson. For example, before tackling a real problem with new facilities, students must be given some time to "explore" the new tools to familiarize with the interface and the corresponding tasks. This is well-known from literature on use of simulations in teaching, but advising instructors is easier based on observations rather than depending upon third party publications to convey the recommendations.

4. **TEACHING PRACTICES** – *Please indicate if* ***your*** *teaching practices or those of* ***others*** *have changed as a result of your project. If so, in what ways. Do you see these changes as sustainable over time? Why or why not?*

Courses for which instructors have changed their practices:

- DSCI 100: Practices not changed, but all Python instead of all "R"
- ENVR 300: managing the workflow of an in-class activity using OCESE dashboard, now done similarly for in-person class as the online version developed during COVID.
- EOSC112: Use of IPCC Climate Atlas and OCESE dashboards was piloted with input from OCESE, so these aspects of the course have been refined after two terms of experience
- EOSC211: major shift in details, although overarching course learning outcomes have changed little:
  - Content via Jupyter Notebooks was piloted in 2021W but dropped in 2022W
  - Use of Jupyter Notebooks instead of MatLab was piloted in 2021W and will remain the norm.
  - Use of autograding that works with Jupyter notebooks (nbgrader) was piloted in 2021W but dropped in 2022W. The in-house workflow for grading assignments and labs will be retained moving forward.
- EOSC325: was a new course in 2021W. It benefitted from OCESE support related to teaching/learning strategies for refining lessons, managing question sets, and introducing dashboards. Teaching practices were refined slightly in 2022W but the practices established will essentially remain for the foreseeable future.
- EOSC 354: Jupyter notebooks introduced instead of MatLab. This approach to teaching and learning time series analysis will remain.
- EOSC 372: One dashboard introduced in 2021W, revised slightly in 2022W, and will remain a part of the course, assuming the instructing team does not change dramatically.
- EOSC 442: Four Jupyter Notebooks were used for the first time in fall 2022, replacing MatLab exercises. These were used again by a different instructor (teaching assistant) in winter 2023. Both terms went according to expectations, so teaching practices are expected to continue.
- Tactics for using Jupyter Hubs to deliver Jupyter notebooks have seen several iterations because UBC's capacity to provide that service has been slowly improving. See also "Project Sustainment", section 5 below.

5. **PROJECT SUSTAINMENT** – *Please describe the sustainment strategy for the project components. How will this be sustained and potentially expanded (e.g., over the next five years). What* ***challenges*** *do you foresee for project sustainment?*
   - An original premise for this project was that open source resources and practices are more sustainable than special-purpose, or proprietary means and methods. One assumption is that the open source community can continue to improve materials that have been widely adopted. Another assumption is that instructors stay connected and committed to these communities, and keep track of changes or advancements. Not all faculty can, or will engage in this way, but it seems true that interest in, and commitment to opensource practices are growing, especially among graduate and undergraduate students.
   - Having students learn, then use, one main programming environment throughout their degree programs (i.e. Python), makes it more likely that changes made to align with that environment will remain in place.
   - Sustainment of OCESE outcomes presumes future faculty can and will teach using Python. There is widespread agreement within the Department that Python is the language of choice, possibly with one or two courses continuing with other platforms ("R", spreadsheets, or possibly MatLab).
   - Documentation has been produced to support sustainability (i.e. guidelines and tutorials). However, the pace of change for open source resources and practices is rapid, and there are multiple approaches to "standard" procedures. Sustainment will depend on EOAS faculty maintaining awareness of how colleagues are teaching their courses.
   - Sustained use, development and maintenance of dashboards requires time and energy from instructors, teaching assistants or hired students. A few instructors will be keen, but time for development, or funding to support student workers, or both, will be required. Otherwise, dashboard use will decline over the coming few years as code becomes outdated, teaching assignments change, and course priorities evolve.
   - Dashboards were all built using the same code libraries for data manipulation, interactivity and plotting expressly to help ensure maintenance is as straightforward as possible.
   - Clarifying responsibilities for maintaining open source resources of all types must be established. For example, a reliable "chain of command" regarding support for Jupyter Hubs must be both available, and clearly understood by instructing teams and students. This involves commitment at department, faculty and institutional levels.
   - Submitting resources individually or collectively as official Open Education Resources (OERs) is one way of disseminating project deliverables, especially if these resources include materials covering "how to use" and "how to maintain or sustain". OERs have yet to be submitted, but we plan to do this before end of 2023.

Challenges for project sustainment

- Maintaining, improving or adding new dashboards has two technical challenges beyond simply writing a new app in Python. Two challenges are:
  - Open source code libraries continue to evolve, and ensuring that library dependencies are both reproducible and upgradable requires some technical knowledge.
    SOLUTIONS: Use conda lock files to ensure code dependencies (i.e. 'environments') are fixed rather

than open for changing as code libraries evolve over time. See this article for some explanation & tactics.

- Preparing a dashboard for deployment on the dashboard server also requires some additional technical knowledge.
  SOLUTIONS: Documentation is being prepared mostly by Prof. P. Austin summarizing specific tactics for managing the resources and servers necessary for beginners' courses to senior courses requiring custom environments. See the growing collection of "How -To Guides" at https://eoas-ubc.github.io/ . These are anticipated to be completed by end of summer 2023.

- Jupyter Notebook hubs: At least 6 EOAS courses now depend upon stable Jupyter hubs. Initial experiences with UBC's open Jupyter Hub were disappointing and very stressful for instructors. For the 2023W academic session, the Department will be experimenting with third party cloud computing hubs supplied by 2i2c.com. We do hope that this kind of computing support for undergraduate teaching will become stable, reliable and effective within UBC. We expect this to happen, as it is a critical aspect of sustaining the efficiency and effectiveness of teaching and learning for the foreseeable future.
- Departmental culture: There needs to be a more regular and effective approach to keeping Department faculty abreast of teaching resources and teaching. Perhaps an annual afternoon event to share changes and "discoveries" regarding teaching EOAS courses could be established? Without some means of helping faculty stay up to date, it will be difficult to sustain existing and future innovations, and to effectively transfer the tactics and materials from current to future instructors of any given course.
- The Department's commitment to explore & support (financially) optimal solutions for hub technology is another indicator that the project's work will be sustained into the foreseeable future.

OCESE Project work to be carried out after June 2023

- The documentation website will be completed.
- Using dashboards during classroom lessons or activities and for assignments will be demonstrated at Department faculty meetings.
- A dashboard development workshop may be run, if EOAS faculty express interest.
- Dashboard for EOSC 429 will be completed, assuming the instructor can participate.
- Climate-science dashboards involving CMIP6 models & data will be piloted in 2023.
- Jupyter notebooks for 8 lab exercises in EOSC 471 will be completed and used for the first time in 2023.
- A third party JupyterHub provider (2i2c.) will be used by EOSC 211 for 2023W. Results, and the apparent stability of UBC Hubs as of May 2024, will determine whether teaching practices in EOSC211 will revert to use of UBC supplied Jupyter hubs for the 2024W teaching session.

6. DISSEMINATION – *Scholarly activities (e.g., publications, presentations, invited talks, etc.) in which information regarding this project was shared. Include author names, presentation title, date, and presentation forum (e.g., journal, conference name, event). These will be included on the TLEF scholarly output page.*

Scholarly contributions are fewer than we in EOAS would have expected owing to constraints experienced during the COVID pandemic. Most of the following were delivered at relatively informal events within UBC. However, >>**two** were presented as peer-reviewed posters at formal conferences.

1. **>>July 2023**: Jones, F., P. Austin, T. Ivanochko, "Lessons Learned While Implementing Open Source Computational Tools and Practices for Learning Quantitative Earth Sciences", Poster at **Earth Educators Rendezvous 2023** meeting, Pasadena, Calif.
2. **May 2023**: Jones, F., P. Austin, T. Ivanochko, TLEF project summary: Embedding Opensource Computational Tools into the Quantitative Earth Science Specializations. Project progress report (poster).
3. **May 2022**: P. Austin, "Creating Inclusion and Accessibility through Data Science: Challenges and Solutions"; Panel discussion, part of UBC's Celebrate Learning week.
4. **May 2022**: Jones, F., P. Austin, T. Ivanochko, Future-ready computing & quantitative skills; opensource solutions in Earth Science courses. Project progress report (poster).
5. **March 2022**: F. Jones, A short EOAS Dep't news item about dashboards.
6. **>>Dec 2021**: Jones, F., C. Shoof, P. Austin, T. Ivanochko, "Reinvigorating Computational & Quantitative undergraduate Curricula for the Earth, Ocean, Atmospheric, Environmental and Planetary Sciences at UBC" (link may not be permanently available to the public), See also a single HTML page version. **American Geophysical Union** (AGU) annual Fall meeting, poster.
7. **May 2021**: Jones, F., P. Austin, T. Ivanochko, Opensource Computing for Earth Sciences Education: Lessons learned in year 1 of 3. UBC TLEF Showcase poster.
8. **May 2021**: P. Austin, Pangeo Showcase. OCESE: Open source computing for Earth sciences education. Also see the video of the event.
9. **Sept. 2020**: P. Austin, "Building an open-source educational community around executable Jupyterbooks". American Meteorolgical Society: 11th Symposium on Advances in Modeling and Analysis Using Python.
10. **August 2020**: P. Austin, M. Colclough, UBC Jupyter Days, Writing Canvas quizzes with Jupyter
11. **July 2020**: P. Austin, Scipy 2020 "BoF" discussion session.