Flexible, student-centred remote learning for programming skills development

ROWLETT, Peter <http://orcid.org/0000-0003-1917-7458> and CORNER, Alexander

IJMEST

International Journal of Mathematical Education in Science and Technology

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/tmes20

Flexible, student-centred remote learning for programming skills development

Peter Rowlett & Alexander S. Corner

To cite this article: Peter Rowlett & Alexander S. Corner (2022) Flexible, student-centred remote learning for programming skills development, International Journal of Mathematical Education in Science and Technology, 53:3, 619-626, DOI: 10.1080/0020739X.2021.1989067

To link to this article: https://doi.org/10.1080/0020739X.2021.1989067

Published online: 29 Oct 2021.

INTERNATIONAL JOURNAL OF MATHEMATICAL EDUCATION IN SCIENCE AND TECHNOLOGY
2022, VOL. 53, NO. 3, 619–626
https://doi.org/10.1080/0020739X.2021.1989067

Taylor & Francis
Taylor & Francis Group

TAKEAWAYS FROM TEACHING THROUGH A PANDEMIC OPEN ACCESS Check for updates

Flexible, student-centred remote learning for programming skills development

Peter Rowlett and Alexander S. Corner

Department of Engineering and Mathematics, Sheffield Hallam University, Sheffield, UK

ABSTRACT

During the COVID-19 pandemic, the teaching of programming for undergraduate mathematicians was moved online. This was delivered asynchronously, with students working through notes and exercises and asking for help from staff via online messages as needed. Staff delivery time was redirected from content delivery into a formal system of formative assessment, which replaced informal discussion and feedback during in-person classes. Formative tasks were submitted and feedback was provided via GitHub Classroom. Students were broadly positive about the formative feedback system and mixed about the need for live delivery. Formal formative feedback highlighted that students may hold incorrect views about the accuracy of task completion, making formal formative submission an effective use of staff delivery time.

ARTICLE HISTORY
Received 28 May 2021

KEYWORDS
Programming; coding; formative assessment; remote learning

2020 MATHEMATICS SUBJECT CLASSIFICATIONS
97P50; 97P40

1. Introduction

When teaching at UK universities moved suddenly online in March 2020 during the COVID-19 pandemic, there was a need to quickly prepare remote teaching. A little more planning was possible for the 2020/21 academic year running from September 2020 to June 2021.

With teaching that moved online that autumn, there were two forms of delivery dominant in the discussion about how to do this:

- pre-recorded videos followed by live online tutorials for students to get support while completing exercises;
- live online classes offering a mixture of lecturer delivery and student activity.

This paper discusses how this context affected teaching for a second-year optional module in a UK undergraduate mathematics degree, 'Programming with Mathematical Applications', including a decision to offer neither of these delivery modes and instead use module staff time for an enhanced program of formative assessment.

Taras (2005) has formative assessment being an evidence-based judgement accompanied by feedback indicating the gap between a piece of work and a particular standard, and

CONTACT Peter Rowlett p.rowlett@shu.ac.uk

an indication of how the work can be improved to reach that standard. Schoenfeld (2015) has formative assessments as 'examinations or performance opportunities the primary purpose of which is to provide students and teachers with feedback about the student's current state, while there are still opportunities for student improvement'.

By presenting the way this was designed and some reflections on its delivery, we hope to contribute to a discussion about engaging flexibly with remote delivery modes and choosing the right approach for the content being delivered.

2. Module design

In their first year, the students studied 'Mathematical Technology', a compulsory module covering various software packages including a brief introductory section on programming. Some students have other prior programming experience, either from school or from personal interests.

The module has a focus on mathematical aspects of programming and provides experience with multiple languages, covering fundamental programming concepts, graphical interfaces, databases and presenting mathematics on the web (Rowlett, 2020). Teaching is delivered via structured notes giving explanations, examples and exercises. For students who are more experienced with programming, a series of challenge exercises are provided to encourage practice. Pre-COVID-19, class time was arranged with a small amount of delivery highlighting key concepts, but most of the time in class being student-centred, with students working on programming tasks and seeking help and feedback from module staff as required.

The motivations for this individualized approach are:

- with diverse prior knowledge and experience of programming, it is not practical to deliver whole-class content appropriate for all students at once; and,
- programming is a practical skill best learned by doing.

The idea is that students use the available learning resources in a way they feel will best aid their development. In response to the notes and exercises, students are expected to write code and this is intended as the performance opportunity which will prompt a feedback interaction. First, the computer will act on the code the students have written and the response may not be as expected. Second, interaction with module staff might occur, either because the student seeks help or because staff casually enquire about performance. This feedback interaction provides information specifically relating to the task, meaning it is formative feedback with an instructional purpose (Hattie Timperley, 2007). In response to this feedback interaction, students are encouraged to reflect on their coding practice and adjust their learning appropriately.

Students are not, in this module, encouraged to work collaboratively in a formal sense. For example, working with others can lead to a situation where some (perhaps more experienced) students will take over responsibility for some aspects of the work, meaning that others do not benefit from development in these areas. This may be welcome in some areas since learning to work to one's strengths is a useful skill, but in the case of fundamental programming skills, it was felt students would benefit from a more thorough grounding.

Students do work collaboratively in other modules, and of course, informal interaction between peers does take place.

The module is assessed through coursework and an in-depth project. The coursework uses well-defined tasks to test taught content. The project takes place over the last six of 24 teaching weeks and allows students to demonstrate a broader range of skills around designing and managing an individual programming project. During the project, there is no direct teaching; rather the scheduled sessions are an opportunity for students to work independently, seeking help and feedback from module staff as required.

3. COVID-19 disruption in 2019/20

The COVID-19 pandemic and associated restrictions took effect starting in March 2020. A decision was made in the evening of 16th March 2020 that teaching at Sheffield Hallam University would be moved to remote delivery effective the following day, ahead of a national lockdown announced by the Government the following week (Public Health, England, 2020).

At this point, students had been briefed on their individual projects and were beginning these. The lack of new content delivery at this stage in the module was advantageous for the sudden shift to remote delivery. Students were guided through installing software (e.g. Python) on their own machines and so enabled to work on their projects at their own pace. The module staff member was available remotely via email, live text chat or video call to offer feedback and support at student request. A requirement of the project is that students provide progress updates during in-class meetings in weeks two and five of the six-week project, and these meetings were changed to email updates. The first of these updates was received on 20th March 2020, with students who did not participate followed up to see if they needed assistance engaging with the module remotely. Nearly all students submitted projects and marks were comparable to the previous year.

4. Change in delivery for 2020/21

4.1. Principles of teaching approach

Teaching began for the 2020/21 academic year with a new cohort of 18 students taught wholly remotely for this module.

It did not seem sensible to replace the classes which took place in person with pre-recorded videos since this would mean a significant shift away from student-centred activity towards staff content delivery via information transmission. Neither did it seem sensible to arrange classes as shared online video calls, because this would require students to maintain a live connection over the internet (not straightforward for all) and seemed to offer little benefit since class time is mostly spent on individual work. Instead, the module was delivered via a flexible, student-centred remote learning approach.

Notes were provided via the web in the usual way. Classes took place asynchronously, with students working through notes and exercises and asking for help as needed via online tools. Students were given flexibility over when they participated in this, though a weekly class timeslot was maintained where module staff were particularly looking to provide

support for the module. Students were encouraged to write with questions, asking for a short video call if needed.

The asynchronous nature of the learning meant that part of the formative assessment process would be missing. The students would still interact with the computer by writing code and could seek help from module staff if required, though note that the barrier to seeking help is raised – writing an email rather than calling someone from across the room for a chat – and some students are reluctant to begin a formal interaction with staff. The missing part is that staff do not have the opportunity to casually observe the student and informally enquire about their progress. Consequently, a formal system of formative assessment was implemented. Along with the usual notes and exercises, a weekly formative task was set. The aim was to enable staff to view students' code, to assess whether students had understood the week's content and provide a context for a feedback discussion in the case where they had not. This was designed to more effectively prompt formative interaction in remote learning circumstances.

4.2. Practicalities

A system was required for students to receive task instructions and submit their work, and GitHub Classroom (https://classroom.github.com/) was selected for this. GitHub is a website that allows storage and sharing of code based on the version-control system `git`. GitHub Classroom is an education-focused tool that allows a task to be set and distributed to students, and their submissions to be collected.

The use of GitHub for this purpose is not inappropriate for this kind of module, indeed it had been recommended by Jones Megeney (2020) for teaching programming in undergraduate mathematics, albeit in a group work setting. It had not been implemented previously as it seemed to be an extra complication that is not core to the needs of undergraduate mathematics students. The opportunity to use the system to submit code for feedback was felt to sufficiently outweigh the distraction of having to learn to use it.

The first class ran as a live video call, and the focus was the delivery of information on how the class would work and student activities of installing Python and submitting a basic task via GitHub. These activities were necessary for engagement with the rest of the module, so the live video call was felt necessary to provide support and avoid a barrier to engagement. Students worked in individual break-out rooms and once each student had completed both tasks, they left the call.

Thereafter, teaching took this form:

- At the end of each class, the notes for the following week were released and students were emailed. These emails highlighted the key learning objectives of the next week's content, provided any relevant instructions and included any other module news. Emails were sent a week in advance of the class time so that students had the flexibility to work through the next topic at their convenience.
- At the start of the next class, this email was resent as a reminder that the class time was now live for those students who chose to work during the designated time.
- During class, answers and feedback were sent first to students who had already asked questions or submitted work earlier in the week, and then to students working during

(1) Write a function called `closure_date` which works for dates in the current academic year. It should take in a date formatted as YYYY-MM-DD and returns `True` if that date is a University closure day, and `False` otherwise.
(2)
   (a) Import `math` and use it to calculate the value of $\cos(5\pi + 2)$.
   (b) Import `emath` and use it in a loop to print the square roots of $1 + ki$ for $k = 1, \ldots, 5$.
   (c) Import `sympy` and use it to print the 2nd derivative of $\sin(e^x)$.
   (d) Use `sympy` to invert a 3×3 matrix of your choosing.
(3) Make a new Excel Macro-enabled worksheet and save it in this folder. Create a UserForm that collects the following information from the user.
   - Name (using a TextBox);
   - Favourite colour (using a ComboBox);
   - Whether you have read 'Hello World: Being Human in the Age of Algorithms' by Hannah Fry (using 'yes'/'no' OptionButtons).

   Use a CommandButton to trigger inputting the data.
   - If any items are left blank, your program should use a MsgBox to ask the user to fill these in and not store the data.
   - If all items are present, your program should store the submitted information in a table in the Excel spreadsheet and hide the UserForm.
(4) Using the MathsJam database file, write an SQL query that identifies the tweets in the database that are by me which mention the word 'baking'. Submit to this repository a plain text file `results.txt` which contains
   - The SQL query that uniquely identified these tweets.
   - The `tweet_text` of the matched messages.
(5) Write a Python script that outputs an `.html` file containing a HTML+MathJax document which displays an integral with at least one randomised constant.

Figure 1. Examples of formative tasks set.

the class time as the need arose. Students were asked to submit their formative work by one hour after the end of the class, though this deadline was not enforced.

For questions, students were encouraged to email or to make use of the GitHub Gist system (https://gist.github.com/), which allows the sending of snippets of program code without this being garbled by an email system.

A selection of the sixteen tasks set during the teaching year is presented in Figure 1.

Work submitted via GitHub is timestamped and presented to module staff, and feedback can be attached to code that has been submitted. An example of the feedback given via GitHub is shown in Figure 2. Fiksel et al. (2019) describe a similar use of GitHub Classroom in the teaching of computational statistics, but with feedback provided by inline comments in the students' submitted code.

Delivery was to a small group of students. The approach to formative assessment via GitHub would scale with increased staffing in response to increasing student size, since it is based on staff writing individual responses to student submissions. However, GitHub Classroom provides some automated marking features (called 'autograding' by Jones, 2020) which may be advantageously explored for large groups; here we felt numbers were such

Figure 2. Example feedback on code given via GitHub.

that the extra development time in an already-busy year did not outweigh the reduced marking time.

5. Student feedback

An attempt was made to collect student views on the module via a questionnaire after teaching had finished and the projects were submitted. This was approved by the Sheffield Hallam University research ethics process (approval no. ER33259510) and distributed to students, but only four students responded and none filled in the free-text questions, limiting the possible analysis. These responses, along with reflection on more informal feedback, are presented in this section. Other sources of feedback that feed into module staff reflections include emails, comments in response to feedback on GitHub, a course-level personal development portfolio and formal course student feedback meetings.

5.1. Live delivery

Survey responses were mixed about the need for live delivery. One student agreed there should be live delivery of taught content, including even if this meant staff time was used up and formative feedback could not be given; though this same student strongly agreed that submitting weekly work had aided their learning. The other students disagreed that live delivery was needed. No negative feedback was received through formal course feedback processes or via other routes about the delivery approach for this module.

Survey respondents agreed that a weekly live drop-in video call should have been offered to answer questions. This is worth considering, though note that a formal online 'office

hour' video call was offered for each staff member during the second half of the year, with zero take-ups from students.

5.2. Formative weekly work

Most survey respondents reported submitting most weekly work and generally finding this straightforward to do. Survey respondents did not feel they dropped behind because of the self-paced nature of the work. In fact, almost all students submitted work regularly at the start of the year, though the numbers submitting formative work decreased as the module progressed. It would be interesting to know why, though this does not seem out of line with the usual experience elsewhere of reduced take-up of formative opportunities as the pressure of summative work in other modules increases. Note also that students commented informally about the tasks getting harder through each topic, which may explain some disengagement.

Survey respondents agreed that submitting work for feedback aided their learning and felt they were getting enough feedback. Informal student comments throughout the year were similarly positive on this point.

5.3. Alternative delivery circumstances

The survey asked students to imagine taking the module in different contexts: (a) in-person teaching with social distancing requirements (sitting apart while wearing masks); (b) no restrictions at all. Two students felt classes should take place in either circumstance, though interestingly one of these had disagreed that live online delivery should form part of the module this year. A third respondent agreed classes should take place only if there were no restrictions at all. A fourth student disagreed there was a need for in-person classes in either circumstance.

6. Discussion/reflection

The redesign of the module delivery involved shifting staff time from live delivery and informal interaction designed to prompt a formative interaction to a formal formative assessment process. The idea for this change arose from concern that the loss of informal in-class interaction would mean that formative interaction was not effectively prompted, but as this resulted in more student work being thoroughly reviewed flaws in the informal system became apparent. It was clear from reviewing work that students often think they have completed a task successfully when in fact they have made some mistake perhaps undetected by the computer response. Consequently, it appears informal questioning in class about the outcome of exercises is not a reliable method of identifying learning misconceptions or development needs when compared to a review of student work. In some ways, this seems surprising, given that program code will give its own feedback in terms of errors, warnings and incorrect output. We theorize that early learners in programming topics might lack the sophistication to identify subtle issues with the code they have written.

It seems, therefore, that answering questions asynchronously and providing detailed feedback on code is an effective use of staff time for this delivery. It is important to note that not all students engaged with formative work consistently throughout the year, raising

issues of disadvantage for the less-engaged. Although it is based on a small sample size, it is interesting to note that not all students feel that classes are needed for this module even if there are no restrictions. However, it is important to know that some would rather be in class even if the circumstances were not much improved.

7. Takeaways

Novice programmers do not appear to be able to successfully detect subtle issues in the code they have written, meaning that simple exercises do not necessarily provide an effective prompt for formative interaction. Therefore formal submission of work for formative feedback is recommended as an effective instructional strategy for programming, even if this means diverting staff time to providing feedback on formative tasks.

It is not clear that an introductory programming module need necessarily take place with in-person classes, even in a teaching situation where classes are usually in person.

Disclosure statement

No potential conflict of interest was reported by the authors.

ORCID

Peter Rowlett http://orcid.org/0000-0003-1917-7458
Alexander S. Corner http://orcid.org/0000-0001-6222-3443

References

Fiksel J., Jager L. R., Hardin J. S., & Taub M. A. (2019). Using GitHub classroom to teach statistics. *Journal of Statistics Education*, *27*(2), 110–119. https://doi.org/10.1080/10691898.2019.1617089

Hattie J., & Timperley H. (2007). The power of feedback. *Review of Educational Research*, *77*(1), 81–112. https://doi.org/10.3102/003465430298487

Jones A. (2020). Introducing autograding for GitHub classroom and the GitHub teacher toolbox.

Jones M. M., & Megeney A. (2020). Programming in groups: developing industry-facing software development skills in the undergraduate mathematics curriculum. *MSOR Connections*, *18*(2), 18–24. https://doi.org/10.21100/msor.v18i2

Public Health, England (2020). The health protection (Coronavirus, Restrictions) (England) Regulations 2020 (SI 2020/350).

Rowlett P. (2020). Programming as a mathematical activity. *MSOR Connections*, *18*(2), 13–17. https://doi.org/10.21100/msor.v18i2

Schoenfeld A. H. (2015). Summative and formative assessments in mathematics supporting the goals of the common core standards. *Theory Into Practice*, *54*(3), 183–194. https://doi.org/10.1080/00405841.2015.1044346

Taras M. (2005). Assessment – summative and formative – some theoretical reflections. *British Journal of Educational Studies*, *53*(4), 466–478. https://doi.org/10.1111/j.1467-8527.2005.00307.x