Teaching Tips/Notes NACTA Journal

Coding Relay: A Collaborative Approach to Teaching Programming

Introduction

The ability to read, write, and architect code is becoming indispensable for students and professionals in Agricultural Sciences to analyze, interpret, and communicate the increasing volume of agricultural data from farming machinery (e.g., yield monitor data), field sensors (e.g., soil moisture probes), and remote sensing information (e.g., vegetation indices). While teaching coding skills using live coding sessions is effective for step-by-step instruction, this method lacks peer interaction. To address this limitation, I introduce the "Coding Relay" approach. Unlike a traditional hackathon, which typically spans several hours or days to a solve complex problem, a Coding Relay focuses on shorter, more focused coding sprints within a classroom setting. In a Coding Relay, each team works in a designated computer where all the code is written, and only one team member codes at a time. The method has two main variations:

- Solo Coding Relay: One student writes code at a time without interacting with other team members, while the rest of the team strategizes and plans together, without seeing the current code progression (e.g., in a different classroom). This variation promotes individual problem-solving skills and the ability to work without immediate feedback. This variation is more suitable for larger teams since there is only one student coding at any given time in the designated computer.
- Cooperative Coding Relay: One student writes code while the rest of the team can see the code progression and can offer support by searching documentation or suggesting alternative coding approaches. This variation encourages teamwork and shared learning experiences. This variation works best for groups of three members, allowing the coder to be in the center and the other two team members positioned on either side.

Procedure

To implement a Coding Relay challenge in a classroom setting an educator would need a designated laptop or desktop computer for each team and a visible timer that resets at specific intervals (e.g., 10 minutes) for changing the coder. The suggested duration for the entire activity is between 50 and 120 minutes depending on the complexity of the challenge and the skill level of the students.

- Step 1: In advance, prepare the coding challenge suitable for the course level and duration. This could be done using a Jupyter Notebook (Kluyver et al., 2016) that includes a brief description of the challenge, the rules of the challenge, hints, and the expected outcome of the code. For easier and standardized grading, it is best

to partition the coding problem into code cells, where students need to add specific code, like importing modules, reading datasets, fitting a model, and creating a figure.

- Step 2: Divide students into groups of 2-5 members, considering the class size. This division could be done the day of the challenge or in advance. Group assignment can be random, skill-based, or based on student preference.
- Step 3: Assign and present the coding challenge. Briefly, go over the code challenge, rules, expected outcomes, and provide hints if appropriate.
- Step 4: Each group designates one member as the "coder" for the first round. The coder sits at the designated computer. Depending on the variation, the rest of the group sits next to the coder to assist and strategize (cooperative variation) or meets in a different space/room to strategize away from the coder (solo variation).
- Step 5: Allow teams to re-read the challenge, select the order of each student in the rotation, and develop a strategy (e.g., define libraries to use). This short period could be from 1 to 5 minutes depending on the complexity and length of the challenge.
- Step 6: Start the timer. The first coder has a set time interval (e.g., 10 minutes) to start solving the challenge. When the timer elapses, teams are required to change the student coder (e.g., within 10 to 15 seconds). Then, reset and restart the timer to signal the start of a new round. Using a buzzer can assist with signaling the time for changing the student coder.
- Step 7: Repeat the process until the total duration of the activity or end of class period.
- Step 8: Teams need to upload the final code file to a platform like Canvas or Github.

Evaluate each group based on their coding speed, accuracy, and collaboration. Consider incorporating peer-grading to assess the perceived contribution of each team member and enhance the fairness of the final grade. The Coding Relay is a versatile teaching tool that can be adapted to various programming languages, skill levels, and class sizes. It provides students with hands-on experience, promotes peer interaction, enhances problem-solving skills, and enables educators to create a dynamic and engaging learning environment that prepares students for real-world programming challenges. This approach is also well-suited for use in competitive events.

Preliminary assessment

The Cooperative Coding Relay variation was implemented as a mid-term exam during the Spring 2024 semester as part of the graduate-level course AGRON 845: Introduction to Scientific Programming and Reproducible Research offered in the Department of Agronomy at Kansas State University. The class was comprised of 13 students that were divided into four groups of randomly assigned students, with three groups consisting of three students and one group consisting of four students.

The challenge was provided in the form of a Jupyter Notebook and consisted of analyzing a dataset of volumetric water content and bulk density levels obtained with the Proctor soil compaction test for a silt loam soil in Tribune, KS (data obtained from Blanco-Canqui et al., 2010). Teams were tasked with importing the Python modules of their choice, fitting a polynomial model, determining the critical moisture content at which maximum bulk density is achieved, creating a publication-quality figure, and briefly explaining the significance of their findings. Each of these tasks were structured as individual code cells in the Jupyter Notebook.

All team members had access to the Jupyter Notebook, allowing them to try alternative lines of code to assist the coder and prepare for their own turn. The challenge had a 50-minute time limit, corresponding to the class duration, with relay intervals set to 10 minutes. Students were permitted to access content from lectures and the official documentation of Python modules but were not allowed to use search engines or generative artificial intelligence language models (e.g., ChatGPT). When done, students submitted the Jupyter Notebook as an assignment to Canvas.

Teams demonstrated strong and sustained engagement throughout the challenge. Teams with members having strong leadership and interpersonal skills were able to quickly develop a strategy and delegate tasks effectively. Teams with inadequate leadership and mixed coding styles tended to introduce unnecessary lines of code, resulting in clutter and increased time spent on debugging the code.

A follow up anonymous survey where students had to briefly describe their experience during the midterm exam using the Coding Relay approach showed that students highly valued the collaborative nature of the exam experience, emphasizing teamwork and shared problem-solving as key strengths. Students found the format engaging and enjoyable, describing it as one of the most fun experiences in their academic journey.

References

Blanco-Canqui, H., Stone, L. R., Schlegel, A. J., Benjamin, J. G., Vigil, M. F., & Stahlman, P. W. (2010). Continuous cropping systems reduce near-surface maximum compaction in no-till soils. Agronomy journal, 102(4), 1217-1225. https://doi.org/10.2134/agronj2010.0113

Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B., Bussonnier, M., Frederic, J., ... & Willing, C. (2016). Jupyter Notebooks–a publishing format for reproducible computational workflows. In Positioning and power in academic publishing: Players, agents and agendas (pp. 87-90). IOS press. https://doi.org/10.3233/978-1-61499-649-1-87

Github repository (with examples)

https://github.com/andres-patrignani/coding-relay

Keywords: Python programming, data science, coding relay, collaborative coding, competitive coding

This is contribution 25-012-J from the Kansas Agricultural Experiment Station.

Submitted by:
Andres Patrignani
Department of Agronomy
Kansas State University
Manhattan, Kansas, USA
andrespatrignani@ksu.edu