2023

Exploring The Potential For Scripting With Simulation In Engineering Education – Practical Examples Using Python And Ansys

Susannah COOKE
Ansys, Inc, susannah.cooke@ansys.com

Susan COLEMAN
Ansys, Inc, susan.coleman@ansys.com

James DERRICK
Ansys, Inc, james.derrick@ansys.com

Exploring the potential for scripting with simulation in engineering education – practical examples using Python and Ansys

S. C. Cooke [1]
Ansys
Cambridge, UK
https://orcid.org/0000-0002-1299-2671

S. Coleman
Ansys
Canonsburg, USA

J. Derrick
Ansys
Cambridge, UK
https://orcid.org/0000-0003-3380-5967

Conference Key Areas: Innovative Teaching and Learning Methods, Engagement with Industry and Innovation
Keywords: Simulation, scripting

ABSTRACT

The ability to use scripting tools to harness the power of complex engineering software is not only critical for research and industry, but also offers opportunities for student learning and development. This paper covers two ways in which undergraduate engineering students have been exposed to Ansys simulation tools to be controlled from Python programs. A pilot series of 'CodeFests' have been held in partnership with university engineering departments, offering a fun way for students to engage with Python coding while exploring the power of scripting to optimise or iterate on solutions. These have used the PyMAPDL structural simulation library, leveraging students' existing understanding of mechanical engineering problems to provide a 'way in'. Students tackled simple mechanical challenges, but with a twist – such as an optimisation requirement which would be beyond manual ability to solve in the time available. In parallel, the potential for scripting tools to provide 'lab in a box' type experiences harnessing the most powerful simulation tools has been investigated. A basic prototype to replicate a fluids lab exercise involving a cylinder in a wind tunnel was created inside a Jupyter Lab running Ansys Fluent through the PyFluent library. This provided a simple, customizable way for students to interact with a 'lab' powered by simulation, without needing to teach them the Ansys Fluent interface and controls first. Both these projects show the potential for harnessing simulation power further in engineering education through scripting methods, to engage and empower the engineers of tomorrow.

---

[1] *Corresponding Author*

S. C. Cooke

susannah.cooke@ansys.com

1 INTRODUCTION

1.1 Scripting Tools and Simulation

Many engineering software tools are designed with specific purposes or specific user groups in mind, which can limit their functionality for other user groups or for people who wish to use them in combination with other tools. The ability to use scripting tools with software, however, opens up a much wider variety of possible use cases, and creates opportunities for both research and teaching.

Since 2020 [1], Ansys has been releasing 'PyAnsys' libraries, which are an Open Source set of technologies that allow users to interface with Ansys solvers such as MAPDL (Mechanical), Fluent (Fluids) or AEDT (Electronics) from an external Python environment. This then allows users to connect these solvers and their outputs to the richness of tools in the wider Python ecosystem.

The value of this in a research or commercial engineering environment is clear, and PyAnsys is already being used in research groups to couple Ansys simulation software with optimisation routines, machine learning algorithms and more [2], [3], [4]. However, an opportunity also arises to deploy these tools within the undergraduate engineering curriculum, where other benefits could be found.

1.2 Motivation

As a leading simulation software company supplying software across a wide range of industry sectors, we at Ansys are aware of the needs of students as they enter the work force when it comes to software tool understanding, etc. But there is evidence that there is a large gap between what industry expects from new graduates and the curriculum students are being taught [5], [6], [7]. One skill in particular that has been highlighted as necessary for the next generation to have is experience with industry-level software [5]. Based on understanding of the value coding and simulation have in industry, and the possibilities which arise from combining the two we have been piloting potential academic-industry engagement in this area in two ways: organising 'CodeFests' and investigating lab work using Jupyter Notebooks and simulation.

1.3 Practice Undertaken

This paper explores two different ways of exposing students to scripting tools: firstly, as a tool for them to explore themselves, enhancing their skillset, and secondly, as an enabling tool to allow introductory students to benefit from advanced software they might not otherwise use.

A pilot series of 'CodeFest' events have been held at a small number of universities, giving students team challenges to tackle by writing their own code leveraging Ansys simulation software. The purpose of these events was to expose students to core scripting concepts and let them explore the possibilities when pairing programming with engineering simulation tools.

Separately, a pilot 'lab in a box' has been created which uses PyFluent within a Jupyter Lab environment to replicate a basic, introductory fluid dynamics lab. The

purpose of this tool is to enable early-years students to benefit from industry-standard simulation tools such as Ansys Fluent underpinning an ‘experiment.’

When looked at in one way, these two projects have opposite goals: the first, to expose students to more complexity and the ability to expand their engineering practice through coding; the second, to reduce the complexity of a simulation tool to help early-years students use it. However they both provide insight into the power of combining scripting and simulation in different areas of undergraduate education.

2 METHODOLOGY

2.1 CodeFest Events

There is a reasonably established tradition of ‘hackathon’ type events being used for student teaching, team-building and awareness-raising, as well as the more commercially-focused industry hackathons that have become common [8]. As a new set of software packages, PyAnsys is an ideal tool to introduce in the focused environment of such an event. However, since there was no free choice of which software students could use, these pilot events were advertised as ‘Codefests’ rather than true hackathons. A 2-day event was held at Cornell University in September 2022, followed by a 1-day event at Virginia Tech university in April 2023, with two further events planned in summer 2023.

CodeFest events were planned in coordination with an academic ‘champion’ at the university: a professor in the engineering faculty who felt that these events would benefit their students. The planning process involved not just the event logistics, but engagement with the academic champions in the months before the event to decide which types of challenges had the most valuable learning outcomes, since research on coding events shows that understanding desired outcomes is key to students getting the most out of the events [9].

The CodeFests were advertised to students 2-4 weeks ahead of the event, through a variety of means tailored to the university ecosystem: emails from faculty admin, a slide at the end of lectures to relevant student groups, small flyers for cafeteria tables and similar. In both cases this strategy was successful and over 100 students registered for each of the events. Students who attended were asked to form teams (and encouraged to make those teams cross-disciplinary) to attempt the challenges. Prizes of reasonable value (approx. 100 U.S Dollars per person) were offered to the winning team, with smaller prizes for creative solutions and other outstanding work.

2.2 Jupyter Lab ‘lab in a box’

Jupyter Notebooks have become increasingly popular in education, as they offer a collaborative working environment which can combine text, images, code, web resources and more [10]. In engineering education, they can help to present complex computations in a way that complements traditional equation-focused teaching [11]. We therefore decided to explore this environment as a way to expose students to simulation with appropriate text, images and links to improve their understanding.

The ‘lab in a box’ style Notebook was developed as a project during a hackathon event, by a small team of 4 people over 24 hours. The motivation for creating a ‘lab in a box’ came from the idea of using powerful simulation tools in place of lab experiments, especially where universities may not have the resources for physical lab equipment, or remote learners may not be able to access it. Simulation can provide a ‘real world’ experiment to compare with theory, and there can be significant learnings from the visualisation capabilities in subjects such as fluid dynamics [11]. However, introductory level students who might benefit most from such visualisations are unlikely to have been trained in industry-standard simulation software, since this is time-consuming and often treated as a ‘readiness for industry’ course option in later years.

The project, then, was to attempt a proof of concept using Ansys Fluent with the python libraries available to create an easy-to-use lab, with simple instructions and user interface, powered by Ansys Fluent underneath. This was implemented in Jupyter Lab, which provides additional flexibility to Jupyter Notebooks. In particular, the ability to ‘hide’ code cells alongside the standard markdown cells means that the code running Fluent could be hidden from students taking the lab, but editable by professors or lab assistants if they wanted to add extra aspects to it.

3 RESULTS

3.1 CodeFest Events

Overall, both events held to date have been judged to be a success, with positive feedback from both students attending and the professor champions.

At both events, despite targeting advertising primarily at undergraduate students, we found Masters and PhD students formed a large proportion of the attendees (34% at Cornell, 45% at Virginia Tech), showing that research students see the potential for combining scripting with simulation tools. Students attended from multiple disciplines: Mechanical Engineering, Computer Science/Computer Engineering and Aerospace Engineering dominated representation, but others from disciplines like Electrical Engineering also attended.

3.1.1 Cornell 2022 CodeFest Results

At this first event, student teams were presented with a choice between ‘guided’ challenges – essentially a set of coding exercises to work through with a small project at the end to apply their new knowledge, designed to introduce students to Python and programming – and ‘general’ challenges, which posed a problem, suggested an Ansys solver to use, and then left teams to create their own solutions using the PyAnsys libraries.

This was a 2-day event, and we saw significant drop-off in attendee numbers between the first day (a Friday) with 71 attendees and the second day (Saturday) with only 25 making it to the end of the second day. Those teams that did complete the challenges, however, produced excellent work showing they had mastered the necessary aspects of both Python and Ansys MAPDL to solve the problems we

presented them with. An example of a challenge and a student team output is shown in Figures 1 and 2, showing how the students were able to engage with both UI creation through Python and controlling the MAPDL solver in the background.

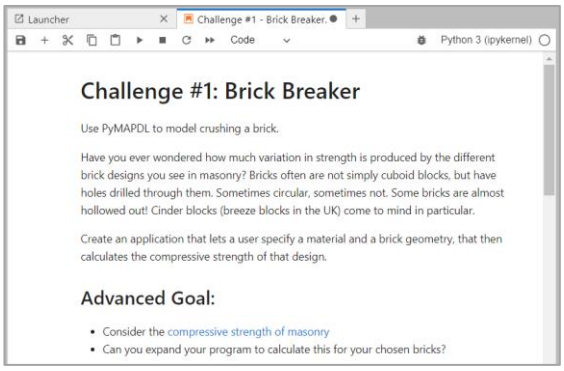

Fig. 1. 'General' challenge as presented to students

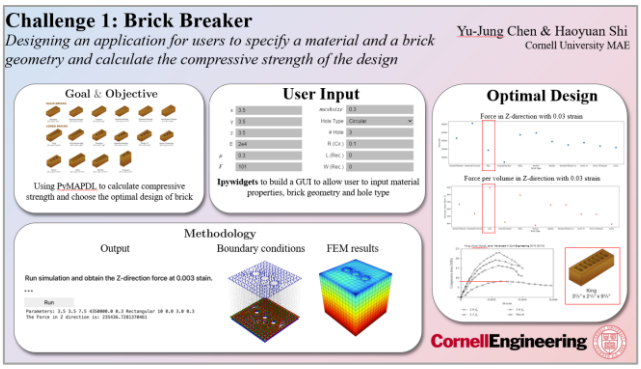

Fig. 2. Final results from one team tackling this challenge

3.1.2 Virginia Tech 2023 CodeFest Results

At the second event, we drew on learnings from the first and offered a new type of challenge which was partly guided and partly open-ended, where students were expected to use coding to find an optimal solution to a simplified 'truss bridge' type problem. This focused more on the coding challenge rather than the engineering challenge, since this was where students had struggled more at the Cornell event, but engineering understanding and validation of the team's solutions was still required for success. We also restricted the event to a single day, running from 9am to 6.30pm, and as a result saw very little drop-off in attendee numbers over the day.

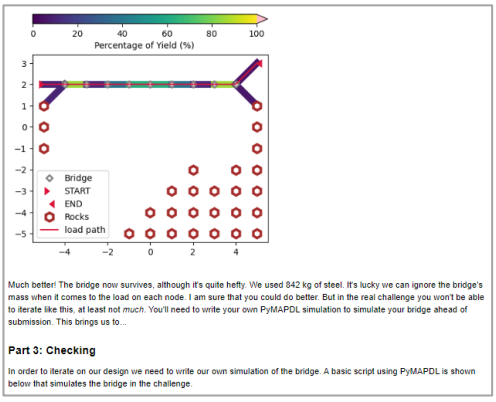

Fig. 3. Team guidance on the bridge challenge

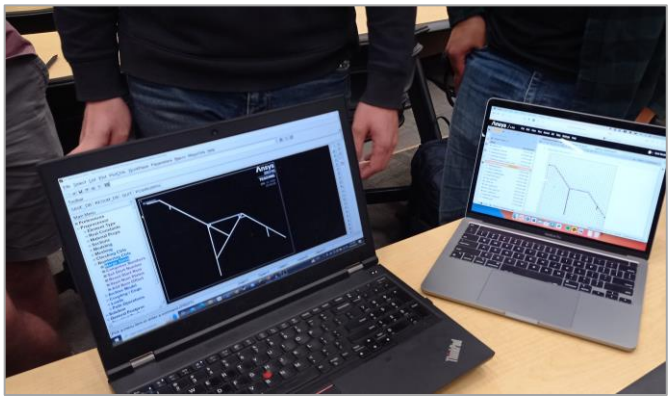

Fig. 4. Bridge optimisation in progress on the computers of a student team

The challenge at this event was made easier to judge by the fact it was a simple optimisation problem – create a bridge across a matrix of nodes, minimise mass without yield failure. Students were encouraged to seek solutions through coding and then test these using PyMAPDL to confirm their behaviour and safety factor. We found that some students struggled to approach the optimisation problem through

coding, preferring instead to sketch out potential design options on paper or whiteboards with engineering calculations done by hand. However, once students engaged with the challenge fully, they were able to harness the power of optimisation and validation. Introductory information for the challenge, and one team's bridge design in progress at the event, are shown in Figures 3 and 4.

3.1.3 Feedback, Lessons Learned and Future Plans

The students who participated in the events and were present at the end were asked for feedback, using an anonymous 'two sticky note' method: one for positive comments, one negative. At the 2-day Cornell event, there was sufficient drop-out on day 2 that this was not representative of most attendees. At the Virginia Tech event, however, this we gathered feedback from >50% of attendees, shown in Table 1.

Table 1: Anonymous feedback from VA Tech Ansys Codefest participants

'What I enjoyed'	**'How could we improve?'**
It was fun. Great staff.	Difficult documentation, wasn't helpful.
This was fun.	Instructions could be clearer, could be more code heavy.
Good food, not "corporate," friendly people, stickers, not too code heavy.	To me, it felt like the deadline is a bit short.
Food is good. Staff seemed sensible.	More detailed documentation would be preferable to get through details.
Impressed with the challenge. It helped me understand the application of Python in solving impossible challenges given the materials and constraints we had to work with this.	I'm not good at social coding. Love code and CAD otherwise. Not a Mechanical Engineer.
Python interface for APDL helps with automation. A good exercise to develop coding skills. Of course, I enjoyed the snacks and goodies.	More robust tutorial on problem. More focus on mechanical analysis portion such as learning about defining geometry and extracting results.
I made new friends. The problem was very interesting.	Finding documentation was a bit tough, process could be easier.
Good food, good swag, friendly staff, staff was nice and helpful.	Problem was too difficult to solve in the time given.
Food is great and great support from the staff.	Would prefer to see each group's designs at the end.
Met people, learned about how MAPDL is integrated into the backend of Ansys.	Problem is challenging for Python beginners. Need a leader of the team to do efficient work, but difficult in rapidly formed team.
Met fun teammates, creative problem solving, got more familiar with Python, had fun, and great networking.	Getting the coordinate was a big challenge, documentation was very difficult, took too much time to figure things out, didn't like the struggle signing in.
Learned a lot, met new people, great food, and great instructors.	Spend first hour explaining problem/example, Difficult for non-programmers.
Liked the format of challenges, the help, food, and the organization of event.	Tutorial on coding – many of us have never done coding.
Super helpful and friendly staff, loved the interactions.	Poor environment, simulations slow, some objectives/processes unclear.
Group work on the challenges, cool/helpful Ansys employees.	Syntax not clear, way too hard to submit answer, rubric unclear.
Team was helpful, challenge was fun to solve and fun to work with my team.	Documentation difficult to parse, incorrect values given initially, thought there would be more Ansys Mechanical, confusion on how to get there.
Format of challenges, comfortable environment, Ansys staff was helpful.	Teach more about Ansys before we start.
Good challenge, really enjoyed.	Demonstrate cases on familiar problems to make sure the programming in environment is understood before we start.
Challenge was fun and an interesting problem learning how to use APDL was more interesting than just Mechanical.	Start of program was too steep of a learning curve, felt more Python than Ansys. We were essentially blind trusting the program. If it ran, great. Need heavy programming skills to be able to push.
Engaging and fun, thanks for making me spend a laze Saturday a better way, helped me get new connections.	Wish there was more instruction on scripting, unclear instructions on actual challenge, wish there was more Ansys Mechanical integration
Whole experience was lovely. Loved to interact with new people and work together as a unit.	Wish there were more Ansys and less Python, based on fliers expecting more of a seminar, documentation needs improved.

Enjoyed how open ended the design process was. Left with a bunch of different ways to approach a problem.	Setup of challenge was frustrating, beams had to be 1 unit which is annoying, could do something more "polybridge."
Very well done overall, learned a lot about environment and scripting within it. Staff and experience was great.	
Intention is good, helpful to understand PyAnsys APIs, networking, fascinating problem and getting used to cooperation.	

Overall, the feedback shows that the social and challenge-related benefits of hackathon-types events were mostly successfully delivered, but there were challenges around coding skills level, particularly for students in engineering disciplines. Teams with computer scientists tended to do better for this reason, although teams of computer science students on their own also struggled, with engineering terminology or fundamental physical understanding. Teams combining both did best. Relatedly, another area requiring improvement, based on the feedback, is the documentation of the PyAnsys libraries, which were originally developed to support expert users of Ansys solvers (those who might already have been scripting in MAPDL, for example). For student users this documentation may need to make fewer assumptions of prior knowledge.

Our academic Champions at the universities were not asked for formal feedback, but both were positive about the events overall and interested in holding more in future. Our VA Tech champion was also interested in the potential to turn our ‘bridge-building’ challenge into a more open-ended teaching tool.

Two further CodeFest events are planned in 2023, at which the technical challenges and support required will be explored further, with one of these events focusing more on postgraduate/research students to see how this affects outcomes and feedback.

3.2 Jupyter Lab ‘Lab in a Box’ Results

The lab use case was based on an early-years fluid dynamics experimental lab run each year in the University of Oxford Engineering department, intended for students who have had lectures on potential flow theory but have not yet been exposed to real fluid flow. It consists of measuring the air pressure at points around a cylinder and showing how the measurements diverge from the predictions of potential flow theory downstream of the cylinder. This allows students to examine the assumptions of potential flow theory through exposure to real, viscous and rotational flow.

Since Fluent simulations can output pressure, they can exactly replicate this lab (minus additional learnings about physical measurement techniques), and potentially add more value through flowfield visualisations, velocity or possibly other variables.

3.2.1 Implementation Details

The Jupyter Lab exercise was thus designed to lead students through a reminder of what potential flow theory predicts, a discussion of the ‘experimental’ setup, and a slider which would allow them to explore different flow speeds spanning laminar and turbulent Reynolds numbers, all without needing them to directly interact with the standard Ansys Fluent user interface which is designed for experienced CFD users. Outputs in Jupyter Lab are in the form of the 2D velocity and pressure fields. Figures

5 and 6 show an example of the Jupyter Lab interface (including UI features such as sliders and buttons implemented using ipywidgets[12]), and the output in Ansys Fluent if it is chosen to show Fluent on screen, or in the Jupyter Lab interface.

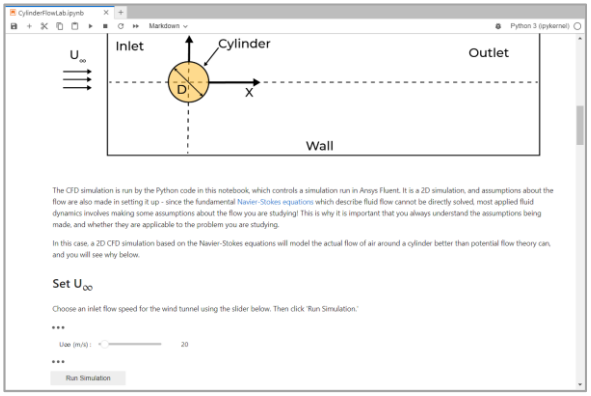

Fig. 5. Jupyter Lab interface with instructions and simple button/slider controls to set values in Fluent and run the simulation

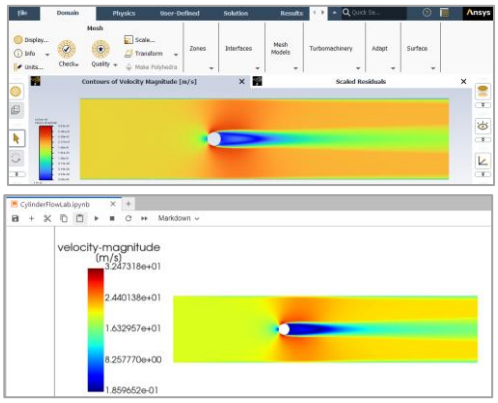

Fig. 6a) and b) Resultant velocity flowfield displayed in a) Ansys Fluent and b) Jupyter Lab, depending on GUI chosen

3.2.2 Lessons Learned and Future Plans

Overall, the proof of concept Jupyter Lab was relatively easy to create – this is something that we could envision professors or student assistants being able to make in future, and customise to their lab requirements.

We plan to share this Jupyter Lab resource in the second half of 2023 with academic users, and get feedback on the ease of use and ease of customisation. We also plan to develop similar resources for other simulation software, for example for mechanical or electronics labs, using PyMAPDL, PyAEDT or other PyAnsys toolkits.

4 SUMMARY AND ACKNOWLEDGEMENTS

In summary, we have found significant interest from students in learning how to couple scripting with simulation tools, for their own development and for their degree work, though there are some inherent challenges to overcome in introducing these through ‘hackathon’ type events, particularly around gauging students’ skills at coding ahead of the event and adapting material appropriately.

We also believe there is a significant opportunity in harnessing the power of scripting, and Python in particular, where simulation tools allow this, to deliver simulation outputs in a format which is accessible to untrained, early-years students. Based on our experience, there are potentially some easy wins in this area for simple use cases where visualisation would strongly support learning outcomes.

We would like to acknowledge the support of Professor Rajesh Bhaskaran at Cornell University and Professor Bob West at Virginia Tech University for partnering with us to deliver the CodeFest events on their campuses. We would also like to acknowledge Dr Christopher Vogel at the University of Oxford for confirming details of the fluid dynamics lab exercise which we attempted to replicate in Jupyter Lab.

REFERENCES

[1] Kaszynski, Alex, et al. 2020. “pyansys: Python Interface to MAPDL and Associated Binary and ASCII Files”. https://zenodo.org/record/4009467#.ZFUW-HbMKF4

[2] Pirkelmann, Simon, Friedrich Raether and Gerhard Seifert. 2022. “Top-down material design of multi-phase ceramics”. *Open Ceramics*, Vol. 9, 100211. https://doi.org/10.1016/j.oceram.2021.100211

[3] Raeisinezhad, Mahsa, Nicholas Pagliocca, Behrad Koohbor and Mitja Trkov. 2021. “Design Optimization of a Pneumatic Soft Robotic Actuator Using Model-Based Optimization and Deep Reinforcement Learning.” *Frontiers in Robotics and AI* Sec. Soft Robotics Vol 8 – 2021 https://doi.org/10.3389/frobt.2021.639102

[4] Phan, Tra Nguyen, Jesus Javier Aranda, Bengt Oelmann and Sebastian Bader. 2021. “Design Optimization and Comparison of Cylindrical Electromagnetic Vibration Energy Harvesters.” *Sensors* 2021, 21(23), 7985; https://doi.org/10.3390/s21237985

[5] Spang, David I. 2014. “Curriculum design and assessment to address the industry skills gap.” *2014 ASEE Annual Conference and Exposition*, pp 24.345.1 - 24.345.14. https://doi.org/10.18260/1-2--20236

[6] Alboaouh, Kamel. 2018. “The gap between engineering schools and industry: A strategic initiative.” *2018 Frontiers in Education (FIE) Conference,* pp 1-6. https://doi.org/10.1109/FIE.2018.8659314

[7] Goold, Eileen. 2015. “Engineering students’ perceptions of their preparation for engineering practice.” *The 6th Research in Engineering Education Symposium* https://doi.org/10.21427/5ck7-0d56

[8] Garcia, Manuel B. 2023. “Fostering an Innovation Culture in the Education Sector: A Scoping Review and Bibliometric Analysis of Hackathon Research.” *Innov High Educ* (2023). https://doi.org/10.1007/s10755-023-09651-y

[9] Porras, Jani, Jyaden Khakurel, Jouni Ikonen, Ari Happonen, Antti Knuta, Antti Herala and Olaf Drögehorn. 2018 “Hackathons in software engineering education – lessons learned from a decade of events.” *SEEM '18: Proceedings of the 2nd International Workshop on Software Engineering Education for Millennials, Pages 40–47*. https://doi.org/10.1145/3194779.3194783

[10] Johnson, Jeremiah W. 2020. Benefits and Pitfalls of Jupyter Notebooks in the Classroom *SIGITE '20: Proceedings of the 21st Annual Conference on Information Technology Education October 2020* pp 32–37 https://doi.org/10.1145/3368308.3415397

[11] Castilla, Robert and Marta Peña. 2023. “Jupyter Notebooks for the study of advanced topics in Fluid Mechanics.*” Comput. Appl. Eng. Educ. (2023)*, pp 1– 13. https://doi.org/10.1002/cae.22619

[12] GitHub. n.d. Accessed May 8, 2023. https://github.com/jupyter-widgets/ipywidgets