# Guest Editors' Introduction to CiSE Special Issue on "Jupyter in Computational Science"

Hans Fangohr[1], Massimo Di Pierro[1], and Thomas Kluyver[1]

[1]Affiliation not available

March 18, 2021

## Guest Editors' Introduction

Notebook interfaces – documents combining executable code with output and notes – first became popular as part of computational mathematics software such as Mathematica and Maple. The Jupyter Notebook, which began as part of the IPython project in 2012, is an open source notebook that can be used with a wide range of general-purpose programming languages.

Before notebooks, a scientist working with Python code, for instance, might have used a mixture of script files and code typed into an interactive shell. The shell is good for rapid experimentation, but the code and results are typically transient, and a linear record of everything that was tried would be long and not very clear. The notebook interface combines the convenience of the shell with some of the benefits of saving and editing code in a file, while also incorporating results, including rich output such as plots, in a document that can be shared with others.

The Jupyter Notebook is used through a web browser. Although it is often run locally, on a desktop or a laptop, this design means that it can also be used remotely, so the computation occurs, and the notebook files are saved, on an institutional server, a high performance computing facility or in the cloud. This simplifies access to data and computational power, while also allowing researchers to work without installing any special software on their own computer: specialized research software environments can be provided on the server, and the researcher can access those with a standard web browser from their computer.

These advantages have led to the rapid uptake of Jupyter notebooks in many kinds of research. The articles in this special issue highlight this breadth, with the authors representing various scientific fields. But more importantly, they describe different aspects of using notebooks in practice, in ways that are applicable beyond a single field.

We open this special issue with an invited article by Brian Granger and Fernando Perez – two of the co-founders and leaders of Project Jupyter. Starting from the origins of the project, they introduce the main ideas behind Jupyter notebooks, and explore the question of why Jupyter notebooks have been so useful to such a wide range of users. They have three key messages. The first is that Notebooks are centered around the humans using them and building knowledge with them. Next, notebooks provide a write-eval-think loop that lets the user have a conversation with the computer and the system under study, which can be turned into a persistent narrative of computational exploration. The third idea is that Project Jupyter is more than software: it is a community that is nourished deliberately by its members and leaders.

The following five articles in this special issue illustrate the key features of Project Jupyter effectively. They show us a small sample of where researchers can go when empowered by the tool, and represent a range of scientific domains.

Stephanie Juneau *et al.* describe how Jupyter has been used to 'bring the compute to the data' in astrophysics, allowing geographically distributed teams to work efficiently on large datasets. Their platform is also used for education & training, including giving school students a realistic taste of modern science.

Ryan Abernathey *et al.* , of the Pangeo project, present a similar scenario with a focus on data from the geosciences. They have enabled analysis of big datasets on public cloud platforms, facilitating a more widely accessible 'pay as you go' style of analysis without the high fixed costs of buying and setting up powerful computing and storage hardware. Their discussion of best practices includes details of the different data formats required for efficient access to data in cloud object stores rather than local filesystems.

Marijan Beg *et al.* describe features of Jupyter notebooks and Project Jupyter that help scientists make their research reproducible. In particular, the work focuses on the use of computer simulation and mathematical experiments for research. The self-documenting qualities of the notebook—where the response to a code cell can be archived in the notebook—is an important aspect. The paper addresses wider questions, including use of legacy computational tools, exploitation of HPC resources, and creation of executable notebooks to accompany publications.

Blaine Mooers describes the use of a snippet library in the context of molecular structure visualization. Using a Python interface, the PyMOL visualization application can be driven through commands to visualize molecular structures such as proteins and nucleic acids. By using those commands from the Jupyter notebook, a reproducible record of analysis and visualizations can be created. The paper focuses on making this process more user-friendly and efficient by developing a snippet library, which provides a wide selection of pre-composed and commonly used PyMOL commands, as a JupyterLab extension. These commands can be selected via hierarchical pull-down menus rather than having to be typed from memory. The article discusses the benefits of this approach more generally.

Aaron Watters describes a widget that can display 3D objects using webGL, while the back-end processes the scene using a data visualization pipeline. In this case, the front-end takes advantage of the client GPU for visualization of the widget, while the back-end takes advantage of whatever computing resources are accessible to Python.

The articles for this special issue were all invited submissions, in most cases from selected presentations given at JupyterCon in October 2020. Each article was reviewed by three independent reviewers. The guest editors are grateful to Ryan Abernathey, Luca de Alfaro, Hannah Bruce MacDonald, Christopher Cave-Ayland, Mike Croucher, Marco Della Vedova, Michael Donahue, Vidar Fauske, Jeremy Frey, Konrad Hinsen, Alistair Miles, Arik Mitschang, Blaine Mooers, Samual Munday, Chelsea Parlett, Prabhu Ramachandran, John Readey, Petr Škoda and James Tocknell for their work as reviewers, along with other reviewers who preferred not to be named. The article by Brian Granger and Fernando Perez was invited by the editor in chief, and reviewed by the editors of this special issue.

**Hans Fangohr** is currently heading the Computational Science group at the Max Planck Institute for the Structure and Dynamics of Matter in Hamburg, Germany, and is a Professor of Computational Modelling at the University of Southampton, UK. A physicist by training, he received his PhD in Computer Science in 2002. He authored more than 150 scientific articles in computational science and materials modelling, several open source software projects, and a text book on Python for Computational Science and Engineering. Contact him at hans.fangohr@mpsd.mpg.de

**Thomas Kluyver** is currently a software engineer at European XFEL. Since gaining a PhD in plant sciences from the University of Sheffield in 2013, he has been involved in various parts of the open source & scientific computing ecosystems, including the Jupyter & IPython projects. Contact him at thomas.kluyver@xfel.eu

**Massimo Di Pierro** is a Professor of Computer Science at DePaul University. He has a PhD in Theoretical Physics from the University of Southampton and is an expert in Numerical Algorithms, High Performance Computing, and Machine Learning. Massimo is the lead developer of many open source projects including web2py, py4web, and pydal. He has authored more than 70 articles in Physics, Computer Science, and

Finance and has published three books. Contact him at massimo.dipierro@gmail.com