# Haystack – a computational molecular data notebook

## A Research Data Management (RDM) "Green Shoots" Pilots Project Report by Clyde Fare and Michael Bearpark Imperial College London

This project was funded as part of Imperial College's RDM "Green Shoots" Programme. In 2014, the Vice-Provost, Research, approved an allocation of £100K for academically-driven projects to identify and generate exemplars of best practice in RDM, specifically frameworks and prototypes that would comply with key funder RDM policies and the College position. The call for projects outlined that frameworks could be based either on original ideas or integrating existing solutions into the research process, improving its efficacy or the breadth of its usage. There was an expectation that solutions would support open access for data; solutions that supported Open Innovation were strongly encouraged.

Six projects were funded, covering different disciplines, faculties and research areas. The projects ran for six months, finishing at the end of 2014. Project reports were made available in 2015.

For more information on the programme and projects please visit:

http://www3.imperial.ac.uk/researchsupport/rdm/policy/greenshoots

Imperial College London

# Haystack – a computational molecular data notebook

Clyde Fare, Michael Bearpark; Imperial College London

Within computational chemistry, sharing and reuse of data currently lags behind other research fields despite the significant benefit of open data to the advancement of science. In this summer project we extended a working prototype of a computational chemical notebook, making it available for all on github[14]. This notebook enables computational molecular researchers to easily share a curated subset of their results and document how those results were generated.

## *Context*

Research councils increasingly require research data to be made available for projects they fund. Similarly some journals[1] now require authors to make their research data available. This push has led to the development of platforms enabling archiving and sharing of data [2]. There have been attempts to make the codes used in scientific research more open and accessible so that the accuracy of their calculations can be evaluated [3,4] alongside experiments with open-notebook science [5], where all research output is made available as it happens. In the field of computational chemistry the culture of sharing data is not yet established although the Quixote project [8] – an initiative to add metadata and archive calculations to web-accessible databases – is under way.

One of the things missing from the theoretical chemical tool set is a means of including a curated part of the actual research process alongside the published results. A painless way of capturing the process used to create the data, plots and analysis etc. (such that it could be made available alongside a published article without requiring significant additional effort from the authors) would greatly increase the transparency and reproducibility of the published research literature.

In an attempt to work towards this goal we built upon the IPython Notebook [6,7] – a computational notebook based on the Python programming language and two prominent python based electronic structure frameworks: the atomic simulation environment[10] and cclib[11]. Our prototype computational chemical notebook added the following functionality:

- Calculations using mainstream computational chemistry software can be set up.
- Calculations can be submitted to run on a high-performance computing cluster.
- Data from completed calculations can be retrieved and visualised.

The prototype was presented at the PyData 2014 conference ("*Changing the way scientists, engineers, and analysts perceive big data"*) and the Thomas Young Centre for materials simulation and at Euro Scientific Python 2014.

## *Problems*

In order for our prototype to be useful to a wider audience it required modification to make it general to computational chemistry rather than specific to the individual setup used by the Bearpark

group. It also required updating to keep up with changes in its parent project the IPython notebook. The prototype was based on numerous scientific libraries making installation a lengthy and somewhat challenging process; this overhead represented a significant barrier to adoption. A further problem was the inability of a single notebook 'page' to capture all the research that occurs in a project. Thus a means of connecting and sharing a collection of notebook pages in a sensible manner was needed.

### *Approach*

The aforementioned issues separated into three domains: refactoring and updating the prototype, testing to weed out bugs, and adding a project tree feature.

For refactoring we sought to separate functionality that should be general e.g. interfacing with computational clusters, from code specific to our particular choice of quantum chemical package. This would allow others to use their own quantum chemical codes within the notebook. We also aimed to update the prototype to use the latest version of IPython and the molecular visualisation tool to its successor JSMol[12] to solve browser compatibility issues. Finally we aimed to make use of commonly available open source tools[13] to make our code and all its dependencies easy to install on any of the three major operating systems so that anyone with an internet connection could easily and freely obtain and make use of it.

Our main feature enabled by the latest version of IPython was the construction of a means to keep track of an entire project composed of multiple notebook pages. We imagined a project 'tree' (i.e. a linked set of nodes) where each node in the tree corresponded to a notebook page containing a set of calculations and their analysis. The entirety of research involved in a project would be contained within this tree. This project tree would be implemented as an alternative notebook dashboard.

For testing, a UROP student was tasked with completing a simple quantum chemical project using the notebook to annotate and keep track of the calculations he performed. He was also to explore the features of the notebook proposed above and test installation on different systems.

### *Achievements*

We successfully completed updating of our notebook code to use the latest IPython 2.x base and our molecular visualiser to use the JSMol package. We refactored the code allowing calculations performed with the popular Gaussian quantum chemistry package such that it was independent of our particular computational resources and could be used by anyone with access to Gaussian. We further removed dependence of the notebook code on our Gaussian interface allowing any of the many quantum chemistry packages with interfaces defined in the atomic simulation environment to be used.

A project tree view was implemented allowing a collection of notebook pages to be linked together to represent a project. Further, a means of selecting collections of nodes and archiving them for inclusion in a publication was created. Installation was streamlined by constructing recipes using the conda build environment popular in the scientific python community. This makes installing all of the necessary libraries our code depends upon automatic, allowing easy installation on all operating systems. Finally our code base has been made available via github and has a BSD license.

Our UROP student successfully undertook an undergraduate project using the computational chemical notebook. And both a PhD student and a new Msci student are making heavy use of the computational notebook.

During the project period we presented a lightning talk at the scientific python 2014 conference.

## *Lessons learned*

In terms of the development process the technical debt incurred during the construction of the prototype took longer than anticipated to pay off. This is a general feature of software development where a higher pace of development leads to code that is not robust and hence a time debt that needs to be paid to rewrite the code in a more robust form. In our case our prototype builds upon several other tools and libraries that were themselves in development. The rapid pace of development of these libraries slightly complicated our situation. A better approach would have been to decide to stick with particular versions of the libraries whilst refactoring our own code and then choose which libraries to update. Whilst in some way this means more code will need to be rewritten it would have simplified the process.

Observing the working habits of our UROP student during his undergraduate project we saw that there were multiple ways of making use of the chemical notebook. This meant unexpected choices were made, some of which were novel and meant he could perform calculations in an efficient manner. However, some meant the notebooks themselves no longer constituted reproducible research as he had stripped out of them the data needed to reproduce the calculations. The power of the computation tools available reinforces the need for a tight coupling between the tools available to make research reproducible and working habits that promote reproducible research.

A further point is the dependence of our project on understanding the basics of a programming language. Whilst usage of the notebook does not require a high degree of programming skill and the Python language itself is designed to be as readable as possible, some programming like activity is necessary and so consequently there is some overhead to using our tool.

## *References*


1. PLOS one data policy. at <http://www.plosone.org/static/policies.action#sharing>

2. Figshare. at <http://figshare.com/>

3. Software Carpentry. at <http://software-carpentry.org/>

4. Mozilla Science Labs. at <http://mozillascience.org/>

5. Open Notebook Science. at <http://www.nature.com/news/2008/080915/full/455273a.html>

6. Pérez, F. & Granger, B. E. IPython Notebook. at <http://ipython.org/notebook>

7. Pérez, F. & Granger, B. E. IPython: a System for Interactive Scientific Computing. *Comput. Sci. Eng.* **9,** 21–29 (2007).

8. Adams, S. *et al.* The Quixote project: Collaborative and Open Quantum Chemistry data management in the Internet age. *J. Cheminform.* **3,** 38 (2011).

9. Alfred Sloan Foundation Grant. at <http://ipython.org/sloan-grant.html>

10. O'boyle, N. M., Tenderholt, A. L. & Langner, K. M. cclib: A library for package-independent computational chemistry algorithms. *J. Comput. Chem.* **29,** 839–845 (2008).

11. Bahn, S. R. & Jacobsen, K. W. An object-oriented scripting interface to a legacy electronic structure code. *Comput. Sci. Eng.* **4,** 56–66 (2002).

12. Hanson R et al. Jsmol and the next-generation of web-based representation of 3d molecular structure as applied to protopedia. *Israel J. chemistry* **53**, 207-216 (2013)

13. Oliphant, T, Anaconda at <http://conda.pydata.org/>

14. Fare, C cc_notebook at <https://github.com/Clyde-fare/cc_notebook>