Reproducible workflows with Jupyter: case study in materials simulation research

Hans Fangohr[1,2], Min Ragan-Kelley[3], Martin Lang[2,1], Marijan Beg[4,2], Juliette Belin[5], Thomas Kluyver[6], Olexandr Konovalov[7], Nicolas M. Thiéry[8]

1) Max Planck Institute for the Structure and Dynamics of Matter, Hamburg, Germany; 2) University of Southampton, Southampton, United Kingdom; 3) Simula Research Laboratory, Oslo, Norway; 4) Imperial College, London, United Kingdom; 5) Logilab, Paris, France; 6) European XFEL, Schenefeld, Germany; 7) University of St Andrews, St Andrews, United Kingdom; 8) Université Paris-Saclay, Orsay, France

Introduction

- Reproducibility is important as a principle of science
- Reproducibility makes science more efficient: *reproducibility enables re-usability*
- We focus on computational worflows (and assume experimental data is digitised)

Common barriers preventing reproducibility

1. Source code of software used is not available
2. Exact procedure, which led to the results reported in the publication, is not shared
3. Information on the computing environment such as supporting libraries and (if required) code compilation details is not revealed

Proposal for reproducible publications [1]

1. Publish software used in research as open source
   - for example in supplementary material of publication and/or
   - as GitHub repository with persistent archival and DOI from Zenodo
2. Use Jupyter Notebooks to direct the study
   - notebooks encapsulate the exact procedure (sometimes called the "protocol")
   - for the publication, provide notebooks that reproduce central results of the paper when executed: for example one notebook per figure
3. Use a "binder-enabled" repository to define the required software environment
   - defines the software required to execute the notebooks, and enables automatic installation
   - `mybinder.org` provides zero-install cloud-based environments in which notebooks from binder-enabled repositories can be executed.

Example repository for reproducible publication [2]:
`https://github.com/lang-m/2022-paper-multiple-bloch-points`

What is a binder-enabled repository?

- (git) repository, typically hosting notebooks
- containing some files that describe what software is required to execute the notebooks (such as `requirements.txt`, `apt.txt`, `environment.yml`, `Project.toml`, `install.R`, …)
- Binder uses existing conventions to describe software environments; thus there is little overhead to learn how to specify the requirements
- A binder service (such as mybinder.org) can create a software environment (based on the specification) and start a Jupyter Notebook server within that environment, so that the notebooks in the repository can be executed.
- A container with the same environment can be created locally using `repo2docker`

Binder demo repository:
`https://github.com/fangohr/reproducibility-repository-example`

Issues

- Notebook only reproducible if cell execution order is linear from top to bottom [1]
- Sustainable funding of `mybinder.org` service (`https://blog.jupyter.org/mybinder-org-reducing-capacity-c93ccfc6413f`)
- Access and use of large data sets from Binder [3]

References

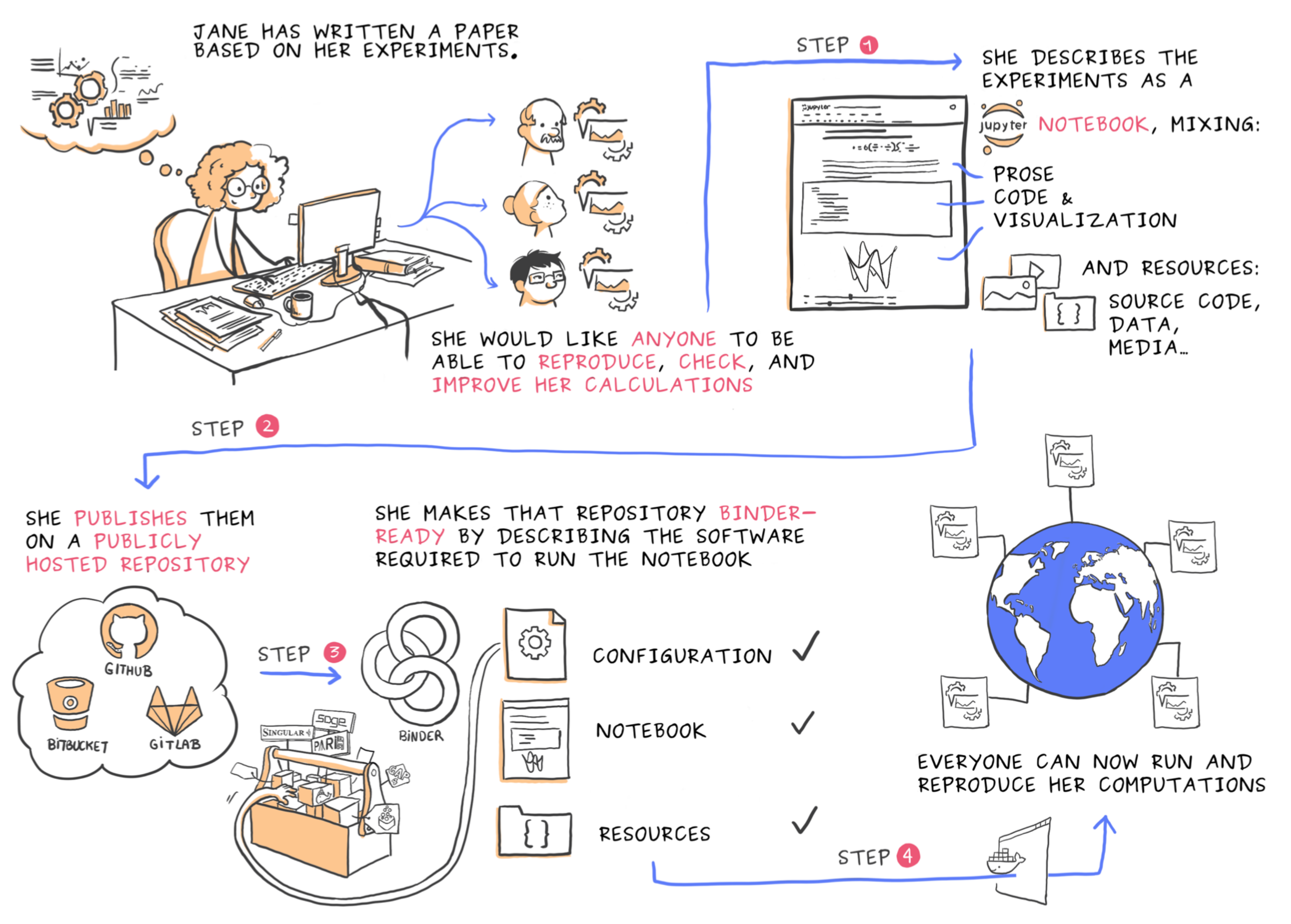
[1] M. Beg, J. Taka, T. Kluyver, A. Konovalov, M. Ragan-Kelley, N. M. Thiery, and H. Fangohr, "Using Jupyter for reproducible scientific workflows," *Computing in Science & Engineering*, vol. 23, pp. 36–46, Mar. 2021. https://doi.org/10.1109/mcse.2021.3052101.

[2] M. Lang, M. Beg, O. Hovorka, and H. Fangohr, "Bloch points in nanostrips," *Scientific Reports*, vol. 13, Apr. 2023. https://doi.org/10.1038/s41598-023-33998-z.

[3] M. Ragan-Kelly, H. Fangohr, N. Thiery, *et al.*, "Supporting open, useful, and reproducible computational environments (SOURCE)," tech. rep., 2022. Grant proposal `https://github.com/minrk/horizon-widera-2022/blob/main/submitted-SOURCE-2022-04-20.pdf`.

[4] M. Beg, M. Lang, and H. Fangohr, "Ubermag: Toward more effective micromagnetic workflows," *IEEE Transactions on Magnetics*, vol. 58, pp. 1–5, Feb. 2022. https://doi.org/10.1109/tmag.2021.3078896.

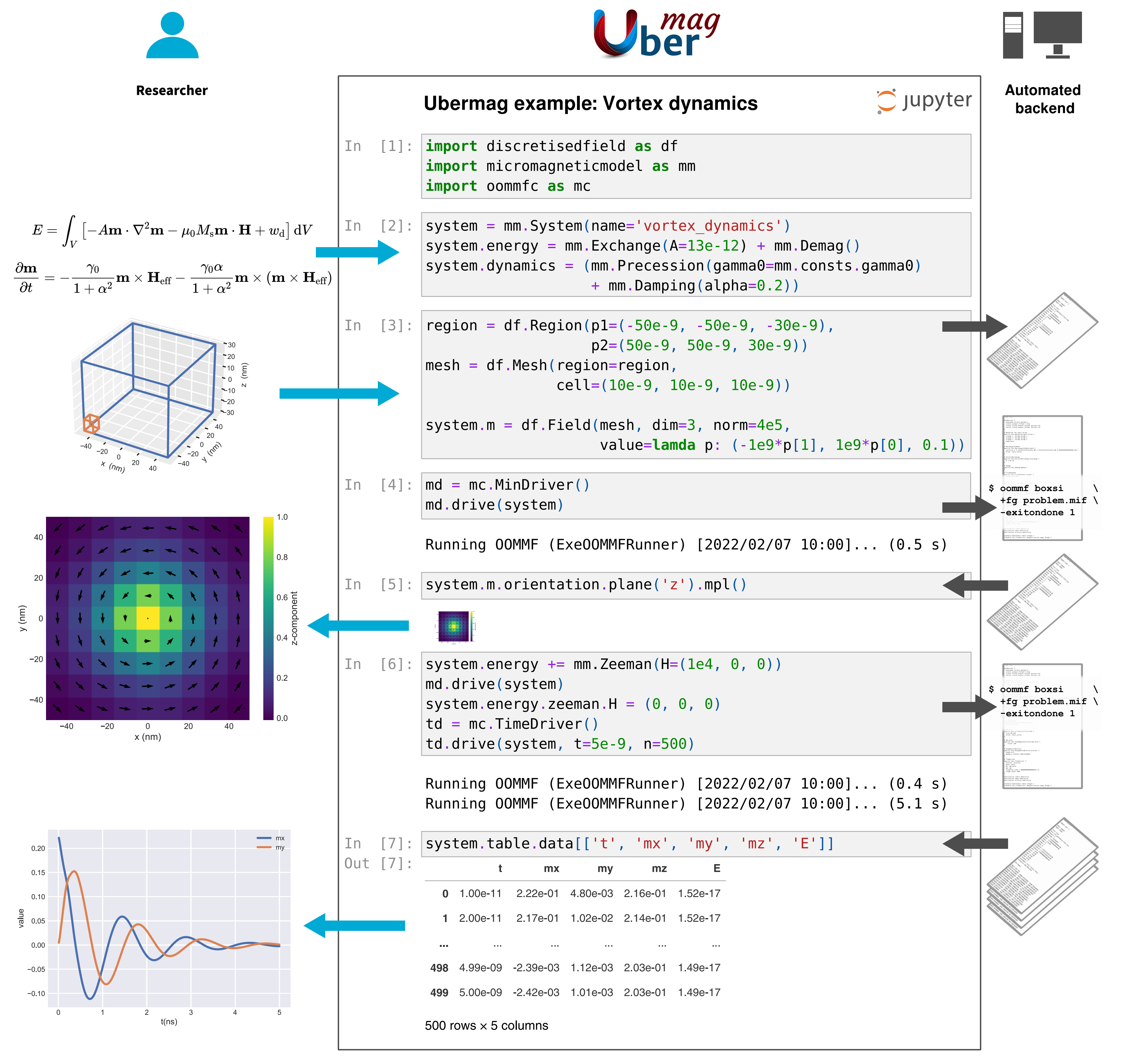
Reproducible publications with Jupyter — summary


JANE HAS WRITTEN A PAPER BASED ON HER EXPERIMENTS.
SHE WOULD LIKE ANYONE TO BE ABLE TO REPRODUCE, CHECK, AND IMPROVE HER CALCULATIONS
STEP 1
SHE DESCRIBES THE EXPERIMENTS AS A jupyter NOTEBOOK, MIXING:
PROSE
CODE &
VISUALIZATION
AND RESOURCES:
SOURCE CODE, DATA, MEDIA…
STEP 2
SHE PUBLISHES THEM ON A PUBLICLY HOSTED REPOSITORY
GITHUB
BITBUCKET
GITLAB
STEP 3
SHE MAKES THAT REPOSITORY BINDER-READY BY DESCRIBING THE SOFTWARE REQUIRED TO RUN THE NOTEBOOK
BINDER
CONFIGURATION ✓
NOTEBOOK ✓
RESOURCES ✓
STEP 4
EVERYONE CAN NOW RUN AND REPRODUCE HER COMPUTATIONS

A case study in which Jupyter Notebooks are used for reproducible data analysis in publications. The software environment is created through the Binder project when required [1].

Reproducible magnetic simulations with Ubermag

- "Ubermag" [4] (`https://ubermag.github.io`) provides a Python interface to (compiled) simulation software (referred to as "backends" below)
- Notebooks provide reproducible transcripts of simulation and analysis



$$E = \int_V \left[-A\mathbf{m}\cdot\nabla^2\mathbf{m} - \mu_0 M_\mathrm{s}\mathbf{m}\cdot\mathbf{H} + w_\mathrm{d}\right]\mathrm{d}V$$

$$\frac{\partial\mathbf{m}}{\partial t} = -\frac{\gamma_0}{1+\alpha^2}\mathbf{m}\times\mathbf{H}_\mathrm{eff} - \frac{\gamma_0\alpha}{1+\alpha^2}\mathbf{m}\times(\mathbf{m}\times\mathbf{H}_\mathrm{eff})$$

Ubermag example: Vortex dynamics

```
In  [1]: import discretisedfield as df
         import micromagneticmodel as mm
         import oommfc as mc

In  [2]: system = mm.System(name='vortex_dynamics')
         system.energy = mm.Exchange(A=13e-12) + mm.Demag()
         system.dynamics = (mm.Precession(gamma0=mm.consts.gamma0)
                            + mm.Damping(alpha=0.2))

In  [3]: region = df.Region(p1=(-50e-9, -50e-9, -30e-9),
                            p2=(50e-9, 50e-9, 30e-9))
         mesh = df.Mesh(region=region,
                        cell=(10e-9, 10e-9, 10e-9))

         system.m = df.Field(mesh, dim=3, norm=4e5,
                             value=lamda p: (-1e9*p[1], 1e9*p[0], 0.1))

In  [4]: md = mc.MinDriver()
         md.drive(system)

         Running OOMMF (ExeOOMMFRunner) [2022/02/07 10:00]... (0.5 s)

In  [5]: system.m.orientation.plane('z').mpl()

In  [6]: system.energy += mm.Zeeman(H=(1e4, 0, 0))
         md.drive(system)
         system.energy.zeeman.H = (0, 0, 0)
         td = mc.TimeDriver()
         td.drive(system, t=5e-9, n=500)

         Running OOMMF (ExeOOMMFRunner) [2022/02/07 10:00]... (0.4 s)
         Running OOMMF (ExeOOMMFRunner) [2022/02/07 10:00]... (5.1 s)

In  [7]: system.table.data[['t', 'mx', 'my', 'mz', 'E']]
Out [7]:
```

	t	mx	my	mz	E
0	1.00e-11	2.22e-01	4.80e-03	2.16e-01	1.52e-17
1	2.00e-11	2.17e-01	1.02e-02	2.14e-01	1.52e-17
...	...	...	...	...	...
498	4.99e-09	-2.39e-03	1.12e-03	2.03e-01	1.49e-17
499	5.00e-09	-2.42e-03	1.01e-03	2.03e-01	1.49e-17

500 rows × 5 columns

```
$ oommf boxsi      \
  +fg problem.mif \
  -exitondone 1
```

Workflow and abstractions introduced by Ubermag for the computation of the dynamics of a magnetic vortex in a ferromagnetic nanostructure. The whole simulation can be driven from a single Jupyter Notebook as shown in the central column. The researcher defines the physical problem (cell 2) and the system to study (cell 3). Both can be visualised directly inside the notebook as indicated on the left-hand side. Ubermag internally communicates with a low-level simulation tool (cells 3, 4 and cell 6) shown as the automated backend in the right column, and returns the simulation results in a high-level format (cells 5 and 7). Plots of the simulation results and further data analysis are part of the notebook. The notebook shown here is a simplified and shortened version of an introductory notebook available at `https://ubermag.github.io/demo.html`.

This work was supported in part by the Horizon 2020 European Research Projects OpenDreamKit (676541) and PaNOSC (823852), and in part by the EPSRC Programme grant on Skyrmionics under Grant EP/N032128/1.

Download from `https://s.gwdg.de/RJxzhq`