OpenDreamKit

Work Package 4: User Interfaces

Overview

The general goals of WP4:

- Improve existing Jupyter tools (T4.2)
- Build new tools in the Jupyter ecosystem (T4.3, 4.8)
- Improve OpenDreamKit components, especially where they interact with each other (T4.1, 4.4, 4.5, 4.12)
- Demonstrate effectiveness in scientific applications (T4.9, 4.11, 4.13)
- Work on Active Documents (T4.6, 4.7)

Overview

Goal: Provide unified interfaces for OpenDreamKit VREs

Broad categories of work:

- **Connect** with underlying computational software
- Notebook interfaces
  - Bring notebooks to **more communities**
  - Improve working with notebooks
  - **Collaborative** workspaces
- **Interactive** documents and documentation

Reporting Period 3

- **No new deliverables**, but software that people use is never complete!
- Instead, focus on supporting, improving work delivered earlier

Milestone M7: Prototype VRE for mathematical researchers

User story: The prototype VRE shall be extended with improved ease of deployment, new functionality such as interactive **3D visualization** and real-time **collaboration**, enabling researchers to collaborate productively in a **shared computational environment**. Finally, integrating notebooks and semantic knowledge into a publication / knowledge system enable a continuous process of leveraging OpenDreamKit components **from research to publication**.

OpenDreamKit produces tools for researchers to use. **WP4 is about making those tools accessible to researchers and research results accessible to others.**

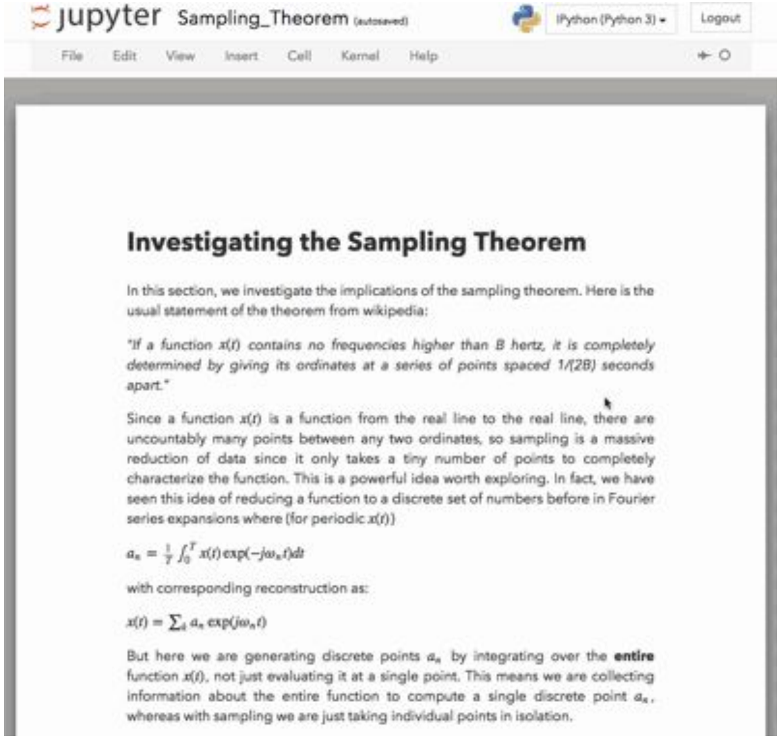
OpenDreamKit bet on Jupyter notebooks

It has paid off!

- **Millions** of notebooks online (over 5M on GitHub alone)
- June 2018, Jupyter awarded prestigious 2017 **ACM Software System Award**
- Previous winners include: UNIX, TCP/IP, the Web, TeX, Java, GCC, LLVM

Background: Jupyter notebooks

Document with code, prose, maths, visualisation



jupyter Sampling_Theorem (autosaved)

Investigating the Sampling Theorem

In this section, we investigate the implications of the sampling theorem. Here is the usual statement of the theorem from wikipedia:

"If a function $x(t)$ contains no frequencies higher than B hertz, it is completely determined by giving its ordinates at a series of points spaced 1/(2B) seconds apart."

Since a function $x(t)$ is a function from the real line to the real line, there are uncountably many points between any two ordinates, so sampling is a massive reduction of data since it only takes a tiny number of points to completely characterize the function. This is a powerful idea worth exploring. In fact, we have seen this idea of reducing a function to a discrete set of numbers before in Fourier series expansions where (for periodic $x(t)$)

$a_n = \frac{1}{T}\int_0^T x(t)\exp(-j\omega_n t)dt$

with corresponding reconstruction as:

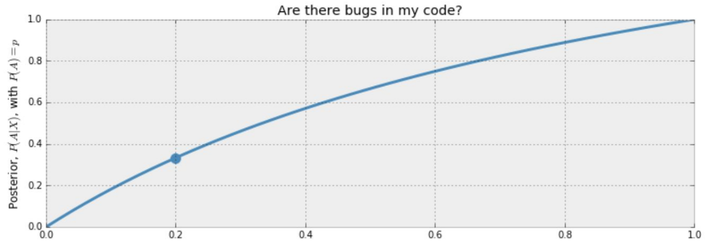
$x(t) = \sum_k a_n \exp(j\omega_n t)$

But here we are generating discrete points a_n by integrating over the **entire** function $x(t)$, not just evaluating it at a single point. This means we are collecting information about the entire function to compute a single discrete point a_n, whereas with sampling we are just taking individual points in isolation.

$$= \frac{2p}{1+p} \tag{11}$$

This is the posterior probability. What does it look like as a function of our prior, $p \in [0, 1]$?

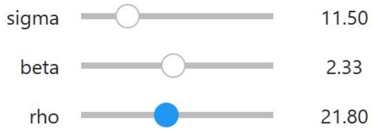
```
figsize(12.5, 4)
p = np.linspace(0, 1, 50)
plt.plot(p, 2*p/(1+p), color="#348ABD", lw=3)
plt.title("Are there bugs in my code?");
```

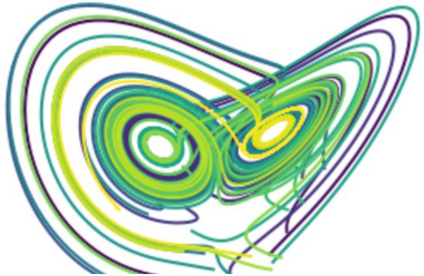


Background: Jupyter notebooks

It also has support for rich, **interactive** UI components:

```
from notebooks.lorenz import solve_lorenz
w=interactive(solve_lorenz,sigma=(0.0,50.0),rho=(0.0,50.0))
w
```



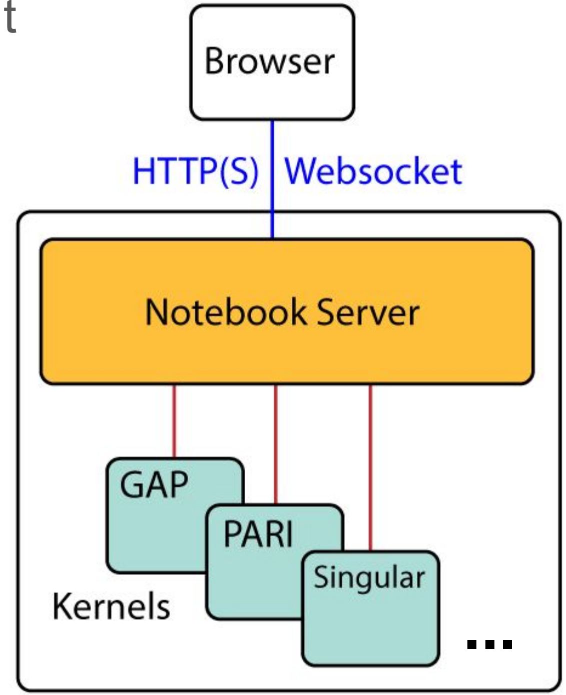


```
In [ ]: @interact
        def factor_xn(n=5):
            display(Eq(x**n-1, factor(x**n-1)))
```

Background: Jupyter notebooks

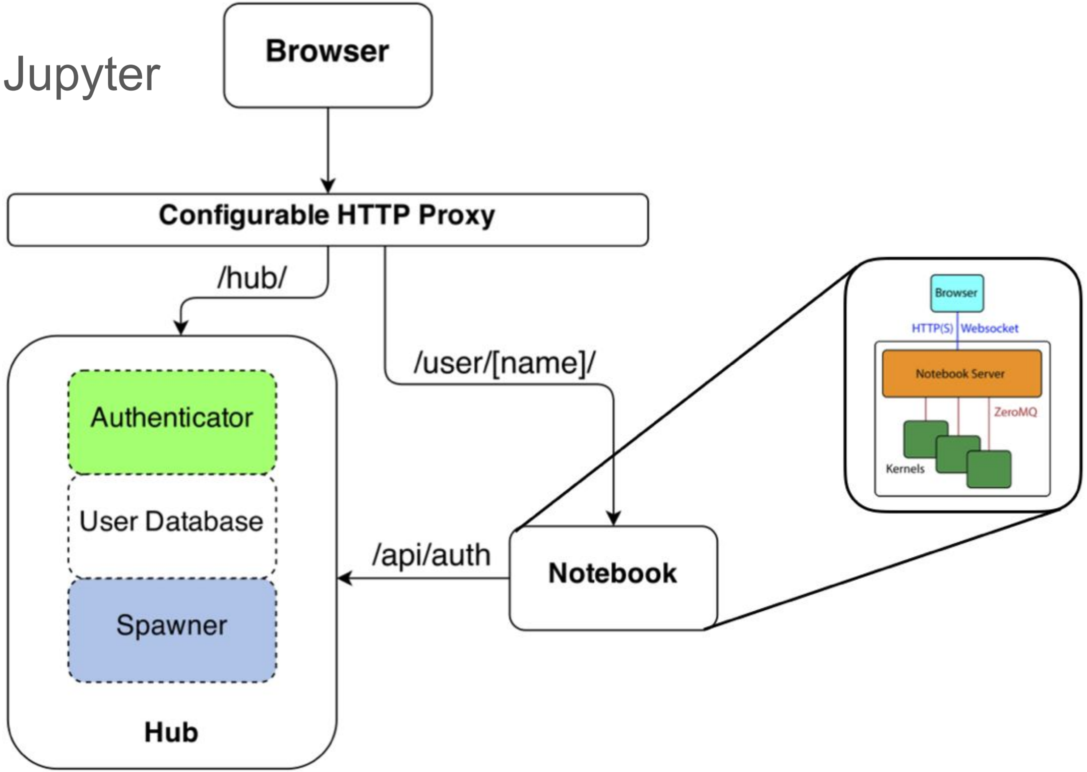
Web-based interactive computing environment

Language-agnostic protocol for computation



Background: JupyterHub

Extensible **VRE** built around Jupyter



Notebook Interfaces

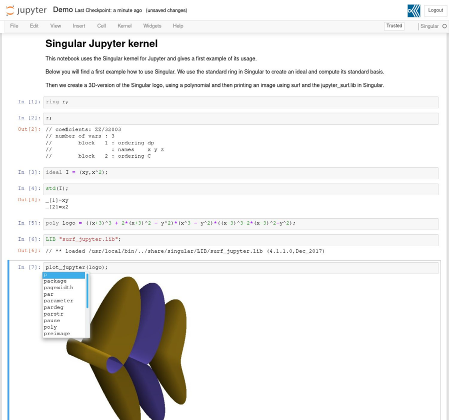
Key areas:

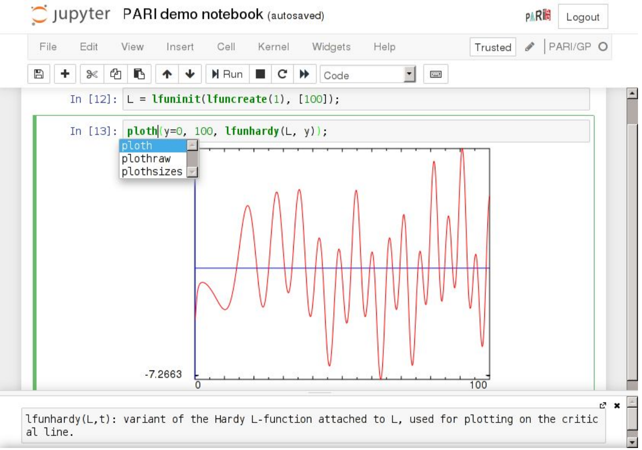
- Improve working ***in*** notebooks
  - Who can use notebooks?
  - What can be done in a notebook?
- Improve working ***with*** notebooks
  - What is difficult or unpleasant or impossible to do with a notebook?

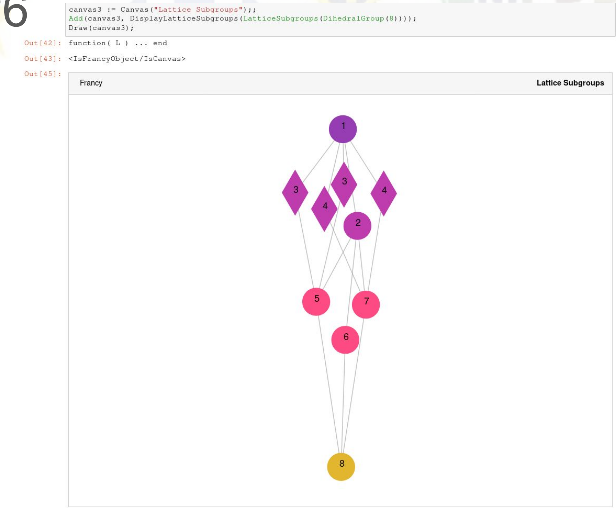
https://opendreamkit.org/try

Highlight: OpenDreamKit kernels

- Now **132 Jupyter kernels**, 8 contributed to by ODK.
- Further improved kernels from first reporting periods (GAP, PARI, Singular). Delivered as D4.7, and MMT, GF, GLF as part of WP6
- Also contributed to kernels: cling (C++), SageMath







KPI: ODK Kernels on GitHub

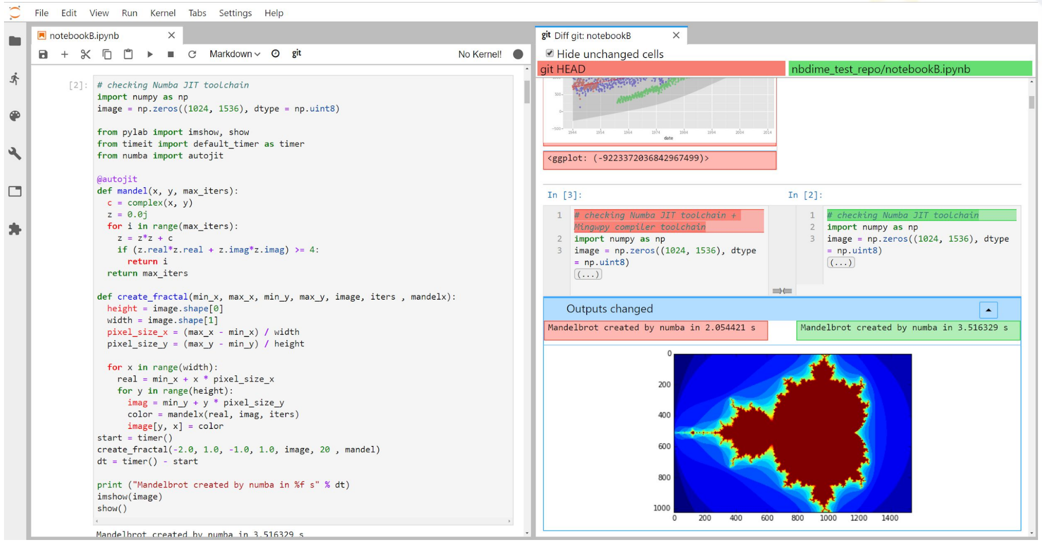
Notebooks found on GitHub using each kernel (code):

- SageMath: 8047
- Xeus-cling: 1216
- GAP: 94
- Singular: 11
- PARI/GP: 5
- MMT, GF, GLF: 5

Year	Total (incl non-ODK)	SageMath	Xeus-cling C++	GAP	Singular	PARI/GP	MMT
2016	467836		0	0	0	0	0
2017	1220829						0
2018	3087257	6199	684	63	8	3	1
2019	5540456	8047	1216	94	11	5	2

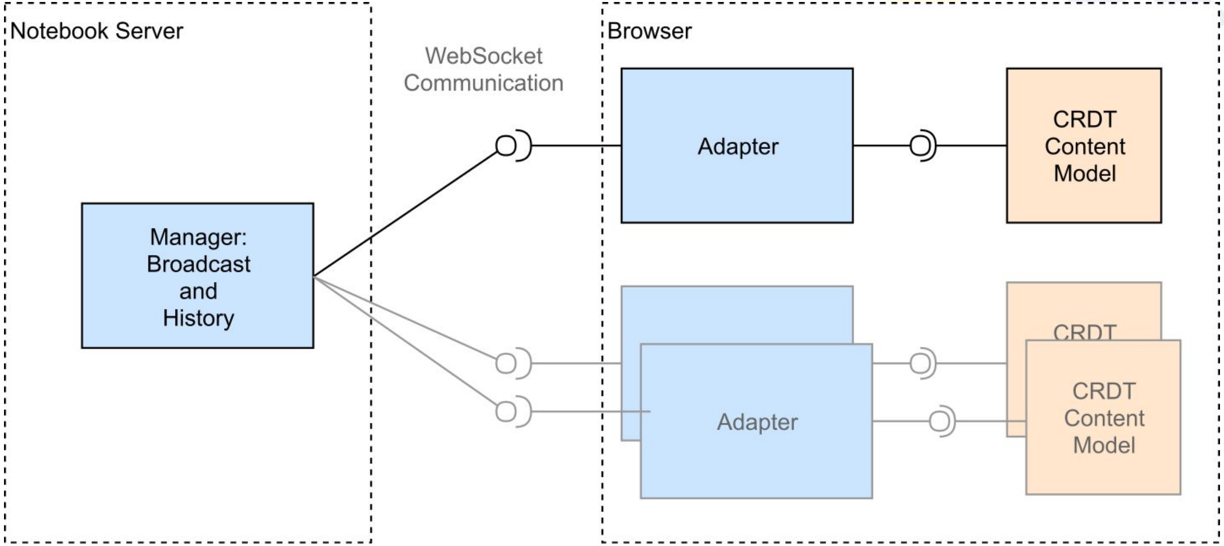
Highlight: nbdime

- Solves **major collaboration challenge** of change review (D4.6). Has been met with enthusiasm, adoption in the community.
- Jupyter Notebook and Jupyter Lab extensions, git integration.

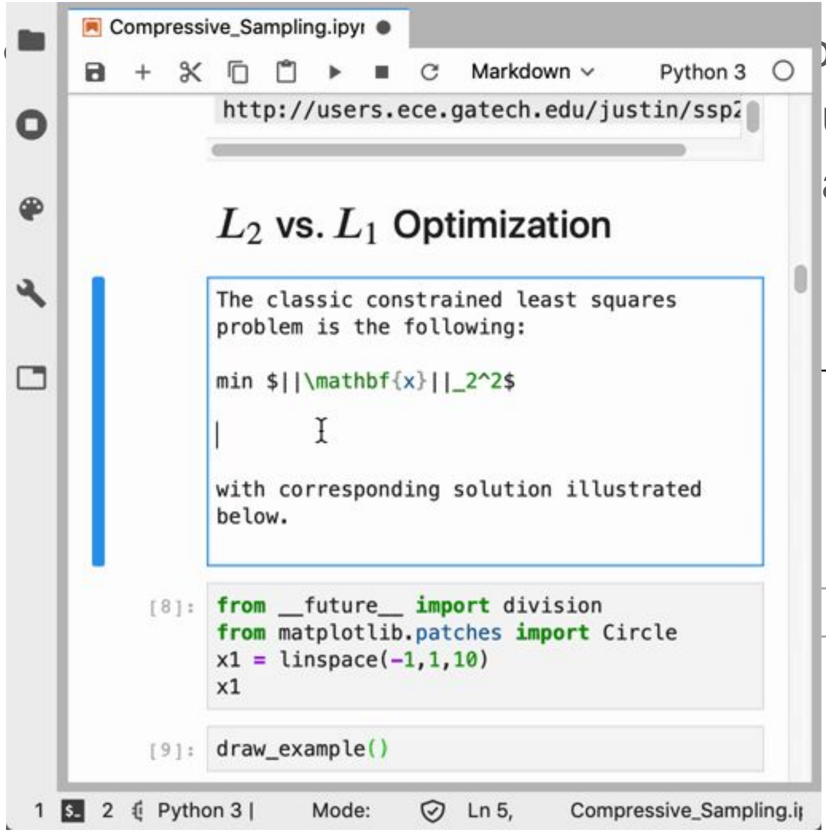


Highlight: real-time collaboration

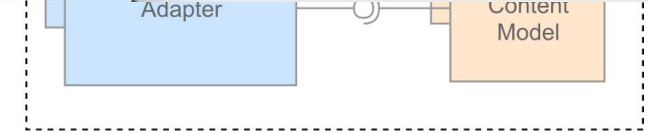
- D4.15 prototype for live collaboration accepted as branch of main JupyterLab repo, continuing development. Required **extensive collaboration and coordination** with the JupyterLab team.

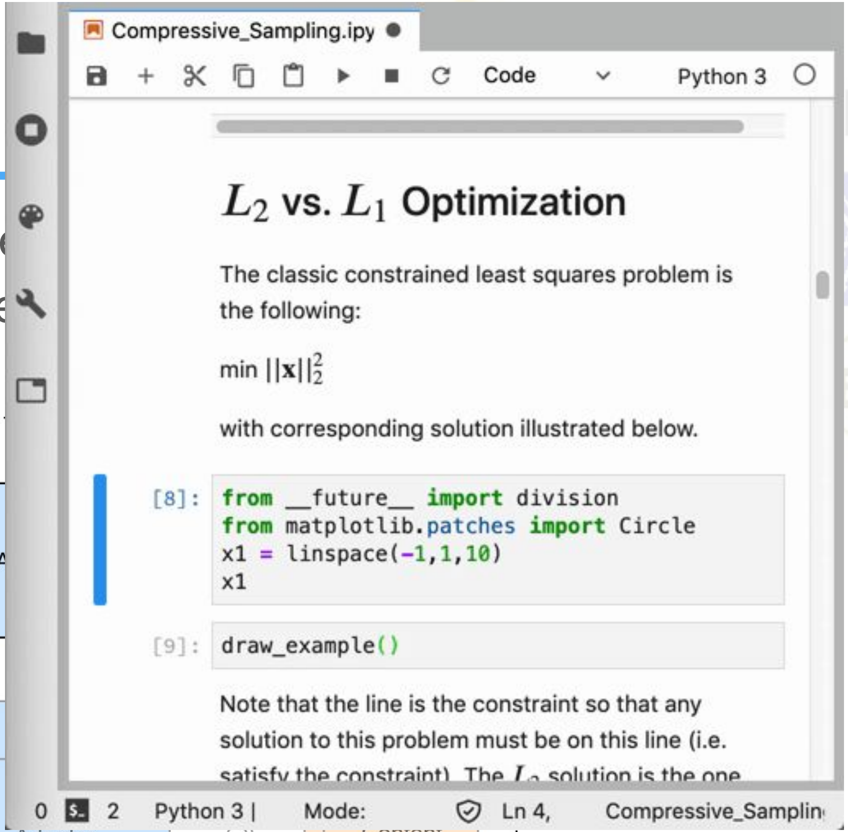


Highlight: real-time collaboration



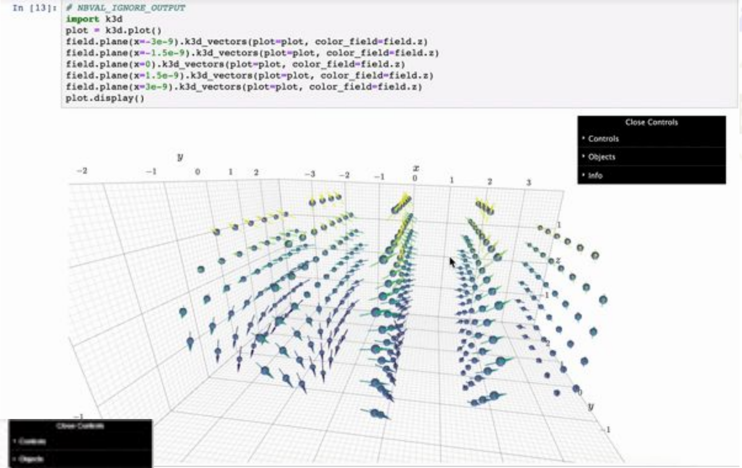
on acc
uired e
am.

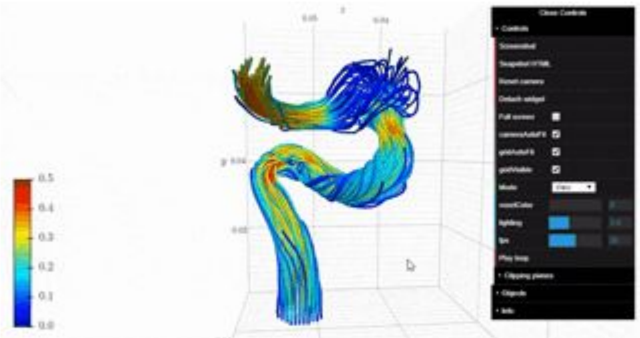


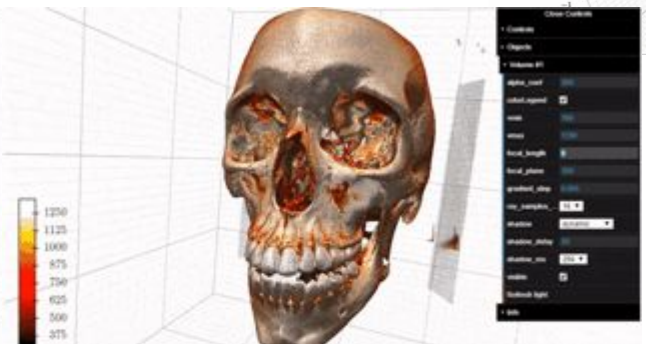


Highlight: 3D visualisation in Jupyter notebooks

- D4.12: Jupyter extension for 3D vis, demonstrated with fluid dynamics, micromagnetics
- Packages:
    - k3d-jupyter
    - ipydatawidgets
    - ipyscales
    - unray
- Solves major challenge for remote VREs







Highlight: Dynamic documentation and exploration system

- Delivered D4.16 (RP2) as two new packages, built on Jupyter widgets for interactively exploring objects in SageMath
    - Sage Combinat Widgets
    - Sage Explorer

```
# A Grid View for a Young tableau
from sage_combinat_widgets import GridViewWidget
t = StandardTableaux(15).random_element()
wt = GridViewWidget(t)
wt
```

1	3	5	6	7	8	11	12
2	10	13					
4	14						
9							
15							

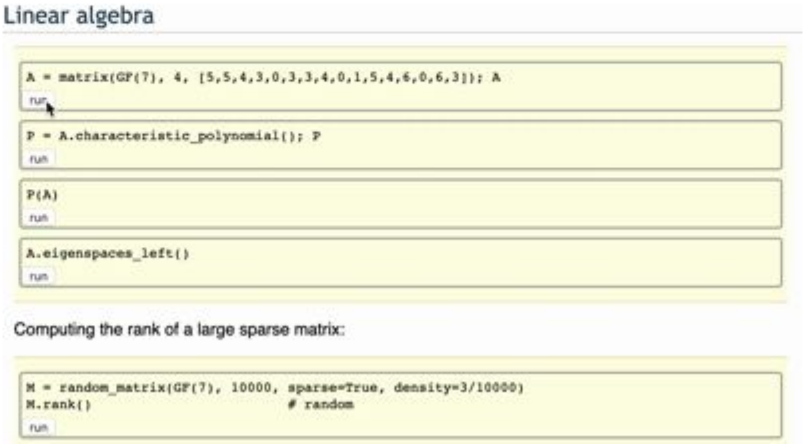
Interactive Documents

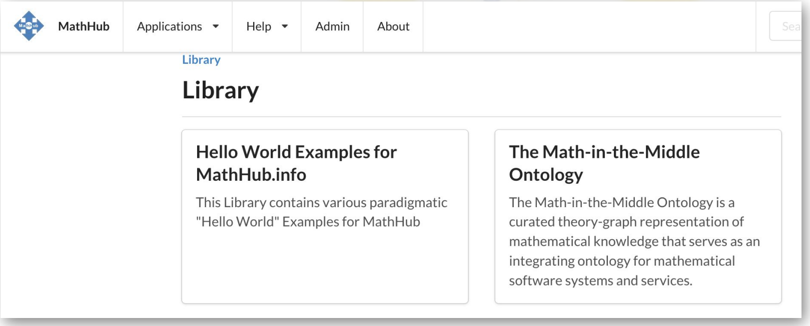
Key areas:

- Active Documents
- Interactive Documentation

Highlight: Active Documents Workshop

Workshop on live documents hosted in Oslo. Resulted in new package: **thebelab**, for embedding execution on any page, integrating tools from JupyterLab and MyBinder.org, and support for notebooks in **MathHub portal**, jupyter.mathub.info

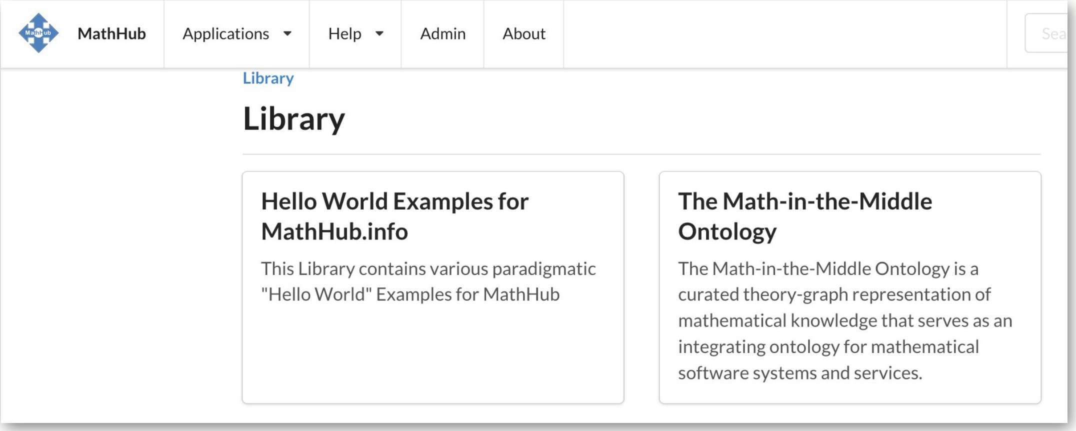




Highlight: MathHub notebook integration

MathHub.info is a portal for active mathematical documents. As part of D4.11, a notebook integration with MathHub was added. This allows:

- Authoring MathHub documents as Notebooks
- **Interactively** exploring existing MathHub documents as a Notebook.



Highlight: Core infrastructure

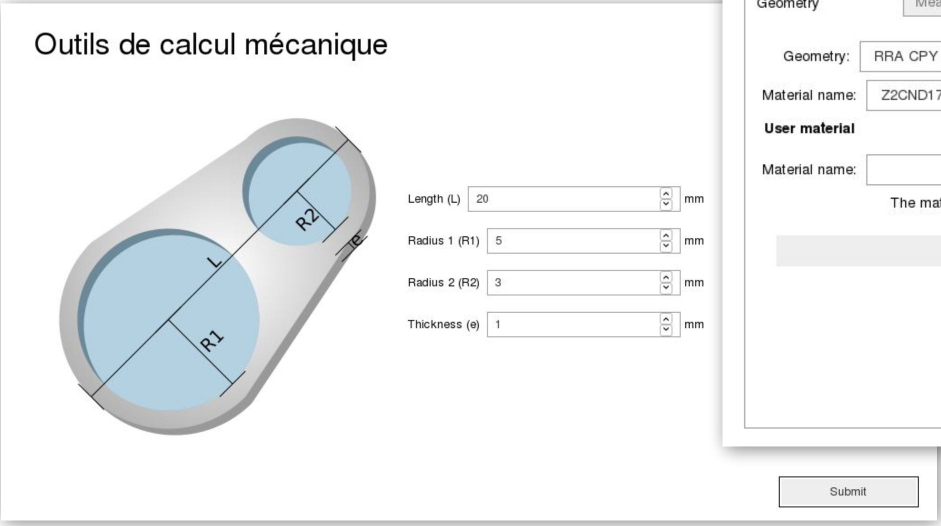
- Refactorisation of SageMath's Sphinx **documentation** system as part of D4.13
- Improve Sphinx support for Cython projects.
  - Enabled building proper documentation for fpylll, CyPari2, CySignals.
- CyPari2 used to be part of SageMath, but it was made a separate package in D4.10 (see also D4.1). This ties into **WP3**.
- To completely enable Cython documentation out of the box, Python needs to be fixed. For this, we submitted PEP (Python Enhancement Proposal) 580. Benefits not only Sage, but **all Python users**.

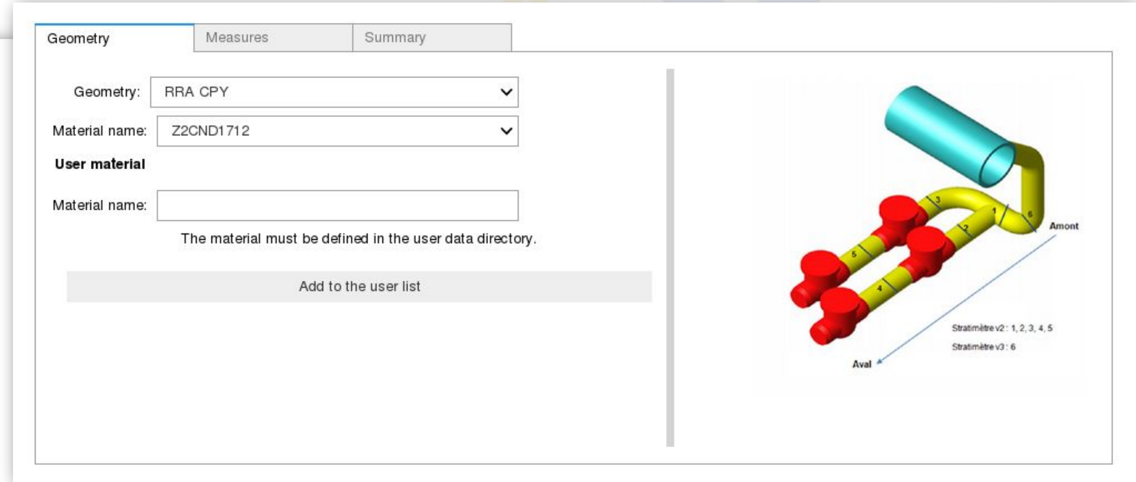
KPI: JupyterHub deployments

- Local CoCalc instance at **Universität Zürich**.
    - Deployed September 2015 - February 2016.
    - People involved: @pdehaye, @williamstein
- Instance of JupyterHub deployed by the Mathrice group
    - Host Infrastructure: France Grille's **LAL cloud**
    - Users: members of math labs in France
    - Main use case: casual use
- Local JupyterHub instance at **Université Paris Sud / Paris Saclay**
    - Host Infrastructure: France Grille's LAL cloud
    - Users: personnel and students of UPSud / Paris Saclay
    - Main use case: use in classroom (Python, Sage, C++), casual use
    - People involved: @VivianePons, @nthiery, @gouarin
- JupyterHub instance deployed on **USheffield**'s HPC system
    - People involved: @mikecroucher
- JupyterHub instance(s) deployed at **UVSQ**
    - Main use case: use in classroom (Sage, Python, C, Apache Spark), casual use
    - People involved: @defeo
- JupyterHub and Binder instances deployed on **EGI** infrastructure, as well as **OVH**, **Azure**, **Google**, and **GESIS**
- Easy deployment of live GAP/SageMath/... notebooks with **Binder**, thanks to the Docker containers (#58); potential alternatives: Debian packaging and Conda packaging.
    - People involved: @nthiery, @minrk, ...
- Local instance of CoCalc (using the Docker container) at the **University of Gent**
    - Main use case: teaching for mathematics students
- Deployed at **jupyter.mathhub.info**
    - With MMT kernel
    - People involved: @tkw1536

Highlight: Simulagora

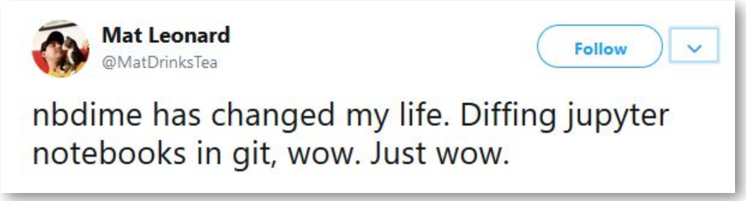
- Logilab **VRE** deployment for application development and deployment.
- Can use **JupyterLab** for application development, which can then be deployed with a simplified parameters form input.





KPI: Usage/impact statistics (since last reporting period)

- nbdime: 1336 stars on github (855 at RP2), 61 contributors (45 in 12 months prior), 388 comments, 107 new issues (94 closed).
- ThebeLab: 96 stars (44 at RP2), 28 contributors (15 in prev 12 months), 124 new issues (12 closed), 292 comments.
- K3D-Jupyter: 154 stars (48 at RP2), 14 contributors, 140 new issues (129 closed), 303 comments.



Summary

- ODK has contributed a **VRE toolkit**
- Jupyter provides the **interface** for these VREs
- Thanks to ODK,
  - **More communities** can use Jupyter
  - SageMath and other ODK software can be used in **more contexts**
  - **Collaboration** in Jupyter is greatly improved
- We have a **demonstration VRE**, built with this toolkit, used for upcoming demos

end

Will present some high-level lessons and future ideas after the demos

Work Package 4

Lessons and Future

Jupyter and the Future

- How did Jupyter contribute to the success of ODK
- Funding determines the direction of project
- Stakeholders are solving their own problems (Banks, large Enterprise)
- Core development still needs support
- EU can ensure that Jupyter core continues to serve Europe, Academia

Jupyter and ODK

- Building on Jupyter has enabled **sharing** and **dissemination** of mathematics research in ODK
- New mathematics communities have **access** to interactive computing and visualisation that didn't before
- **Education** and **research** facilitated by improved interactive capabilities
- Several **major challenges** of coordinating with open source projects with many stakeholders, but **big rewards** in the end, especially for long-term **sustainability**
- Web-based Jupyter makes it easier to build **cloud-based VREs** with ODK

Before OpenDreamKit

- Jupyter showed promise as a **shareable** document and **interactive** environment
- Only available to certain programming communities (Python, Sage, Julia, R, etc.)
- However, there were **key gaps** in the collaboration workflow
- **Cloud/remote** computing growing, but desktop capabilities were lost
- Many opportunities for interactive **explorations**

After OpenDreamKit

- Jupyter has proven valuable for sharing and interactive computing
- Now available to many **more communities** (ODK kernels)
- Gaps in collaborative workflow have been filled (**nbdime**, **nbval**, **JupyterLab**)
- **3d visualization** enabled in Jupyter via **K3D, ipydatawidgets, pythreejs.**
- Improved interactive **exploration** via widgets
- Improvements throughout the Jupyter and Python ecosystems along the way

Jupyter and the Future

- Building on **existing projects** has enabled ODK to have high impact
- **Contributors tend to solve their own problems.** Corporate participation in Jupyter is valuable, but priorities often differ from academia and public good
- **Core project development** still needs long-term funding to be secure
- Funding core development **enables EU to guide direction** of Jupyter or similar projects and **support STEM** research and education through key infrastructure
- ODK is special because it includes **explicit funding for core software infrastructure**. There should be more of this!

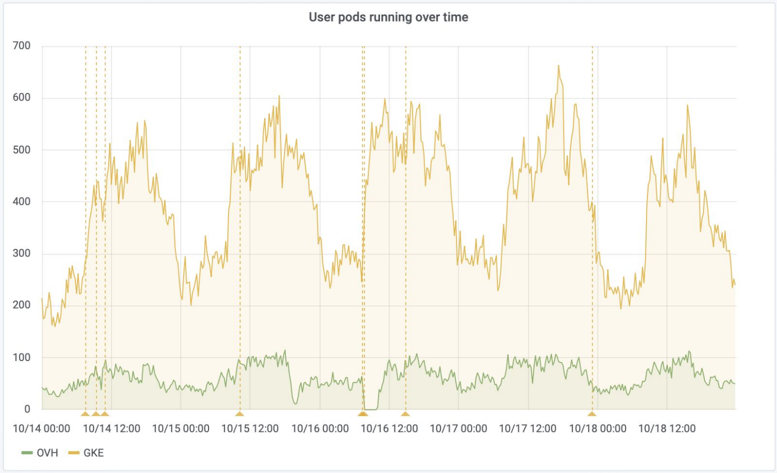
Binder: Jupyter for Open Science

- Submitted BOSSEE proposal to **EOSC** to fund Binder for Open Science (not funded)

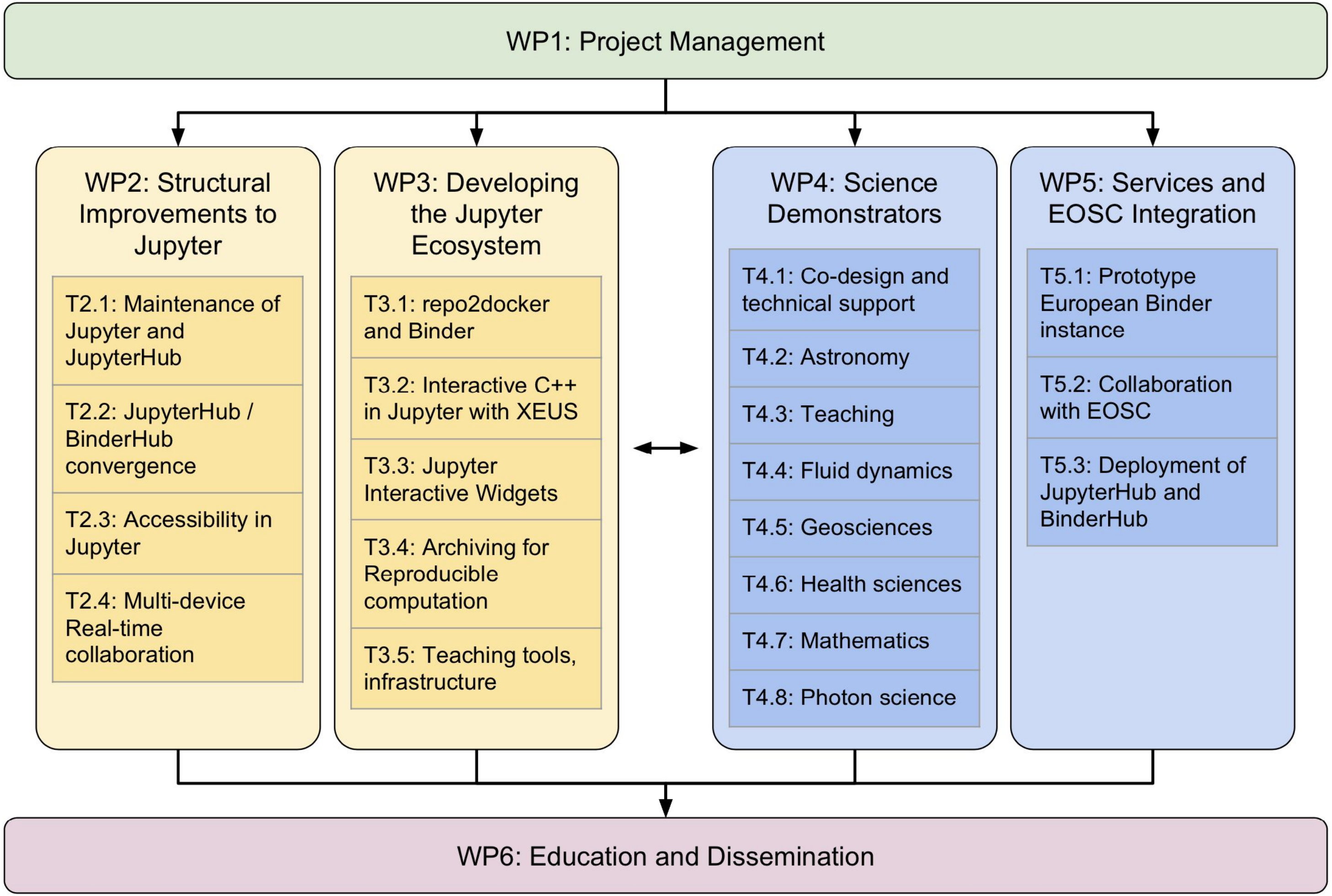
Binder currently hosted by:

- Jupyter Team (US, Google Cloud)
- OVH (France, cloud provider)
- GESIS (Germany, social sciences institute)
- Turing Institute (UK, open science advocate)

Thousands of people are using Binder every day



Binder is the frontier of Jupyter in Open and Reproducible Science


WP1: Project Management
WP2: Structural Improvements to Jupyter
T2.1: Maintenance of Jupyter and JupyterHub
T2.2: JupyterHub / BinderHub convergence
T2.3: Accessibility in Jupyter
T2.4: Multi-device Real-time collaboration
WP3: Developing the Jupyter Ecosystem
T3.1: repo2docker and Binder
T3.2: Interactive C++ in Jupyter with XEUS
T3.3: Jupyter Interactive Widgets
T3.4: Archiving for Reproducible computation
T3.5: Teaching tools, infrastructure
WP4: Science Demonstrators
T4.1: Co-design and technical support
T4.2: Astronomy
T4.3: Teaching
T4.4: Fluid dynamics
T4.5: Geosciences
T4.6: Health sciences
T4.7: Mathematics
T4.8: Photon science
WP5: Services and EOSC Integration
T5.1: Prototype European Binder instance
T5.2: Collaboration with EOSC
T5.3: Deployment of JupyterHub and BinderHub
WP6: Education and Dissemination