

# Provenance and Reproducibility: A Look into Jupyter Notebooks

Dr. Sheeba Samuel

Friedrich Schiller University, Jena, Germany

Thuringian RDM-Days  
22<sup>nd</sup> June 2021



# Agenda

- Introduction to Jupyter Notebooks
- Provenance Information Management in Notebooks
- Reproducibility of Notebooks

# Jupyter Notebook

- A virtual environment for literate programming.
- Share code along with documentation
- *Collaborative creation of reproducible computational narratives that can be used across a wide range of audiences and contexts.* - [Project Jupyter]
- Open-source software
- Data exploration, run simulations and visualization

# Jupyter Ecosystem

- Jupyter Notebook
- JupyterLab
- JupyterHub
- Binder

The screenshot displays the JupyterLab interface. On the left, a sidebar shows a file browser with a list of notebooks: Data.ipynb, Fasta.ipynb, Julia.ipynb, Lorenz.ipynb (selected), R.ipynb, iris.csv, lightning.json, and lorenz.py. Below the file browser are sections for 'Running' (empty), 'Commands' (empty), 'Cell Tools' (empty), and 'Tabs' (empty). The main area is divided into three panes. The top pane shows the notebook content, which includes the Lorenz system equations:  $\dot{x} = \sigma(y - x)$ ,  $\dot{y} = \rho x - y - xz$ , and  $\dot{z} = -\beta z + xy$ . Below the equations, a text prompt says 'Let's call the function once to view the solutions. For this set of parameters, we see the trajectories swirling around two points, called attractors.' The bottom-left pane shows the 'Output View' with a 3D plot of the Lorenz attractor. The bottom-right pane shows the code for the Lorenz system, including a function `solve_lorenz` and a function `lorenz_deriv`. The code is as follows:

```
9 def solve_lorenz(N=10, max_time=4.0, sigma=10.0, beta=8./3, rho=28.0):
10     """Plot a solution to the Lorenz differential equations."""
11     fig = plt.figure()
12     ax = fig.add_axes([0, 0, 1, 1], projection='3d')
13     ax.axis('off')
14
15     # prepare the axes limits
16     ax.set_xlim((-25, 25))
17     ax.set_ylim((-35, 35))
18     ax.set_zlim((5, 55))
19
20     def lorenz_deriv(x,y,z, t0, sigma=sigma, beta=beta, rho=rho):
21         """Compute the time-derivative of a Lorenz system."""
22         x, y, z = x,y,z
23         return [sigma * (y - x), x * (rho - z) - y, x * y - beta * z]
24
25     # Choose random starting points, uniformly distributed from -15 to 15
26     np.random.seed(1)
27     x0 = -15 + 30 * np.random.random((N, 3))
28
```

JupyterLab Interface

# Jupyter Notebooks: Facts

- Formerly known as *IPython* Notebook
- 1.7 million Jupyter notebooks on Github
- Millions of *users*
- Different *computational kernels* including Python, R, and MATLAB
- Export in different *formats* like HTML, LaTeX, PDF

# Structure

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Markdown Cell

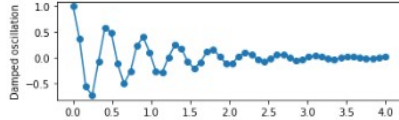
Damped Oscillations: Simple demo with multiple subplots.

Code Cell

```
In [1]: %matplotlib inline
import matplotlib.pyplot as plt
import numpy as np

In [5]: x1 = np.linspace(0.0, 4.0)
y1 = np.cos(20 * np.pi * x1) * np.exp(-x1)
plt.subplot(2, 1, 1)
plt.plot(x1, y1, 'o-')
plt.ylabel('Damped oscillation')
plt.savefig("t1.png")
plt.show()
```

Output



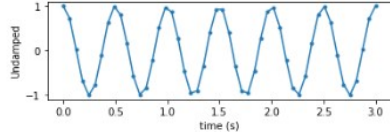
Raw Cell

Another subplot with change in parameter

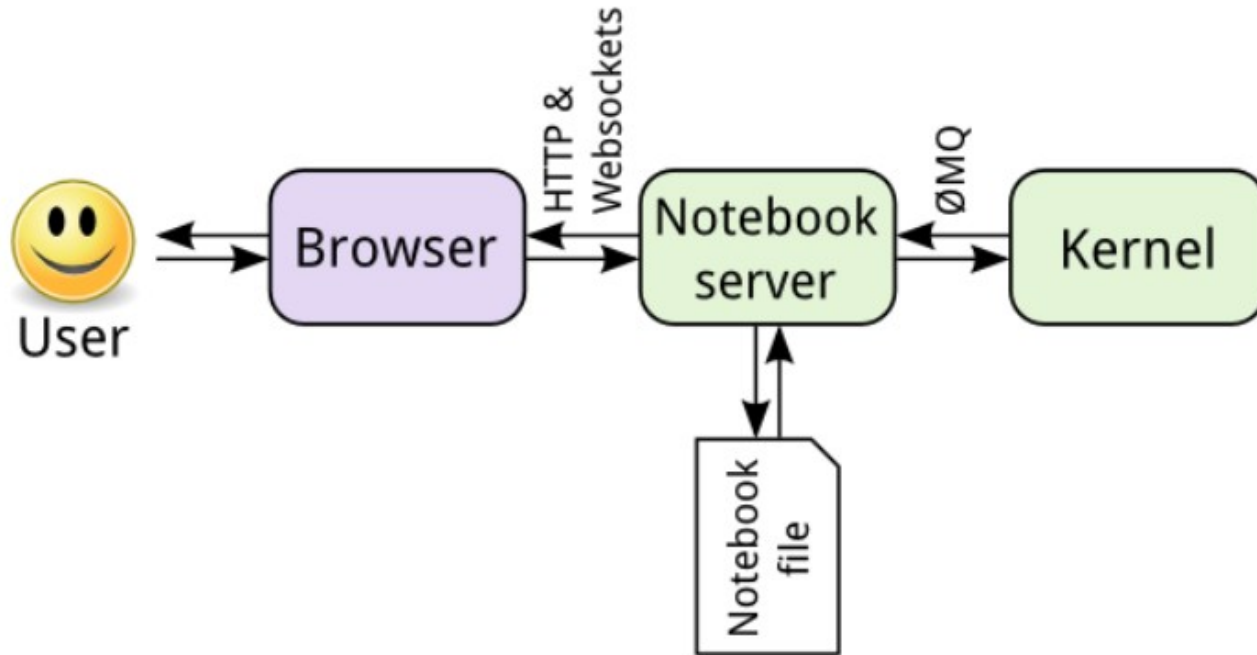
```
In [3]: x2 = np.linspace(0.0, 3.0)
y2 = np.cos(4 * np.pi * x2)

In [4]: plt.subplot(2, 1, 2)
plt.plot(x2, y2, '-.-')
plt.xlabel('time (s)')
plt.ylabel('Undamped')
plt.show()
```

Execution Count



# Architecture

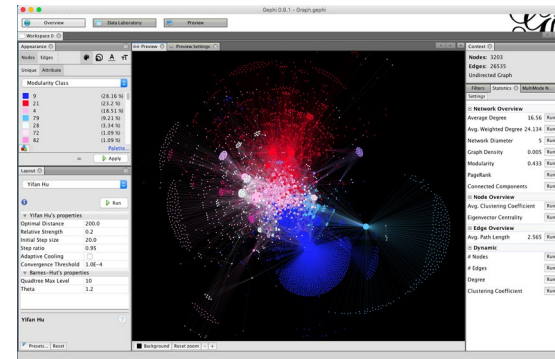
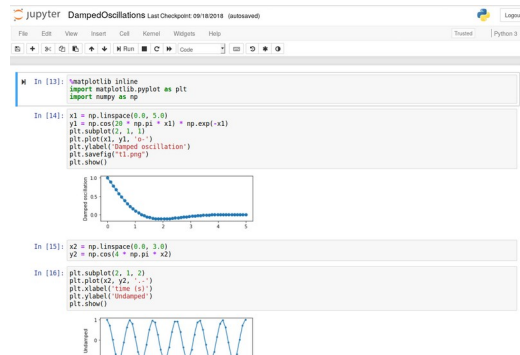
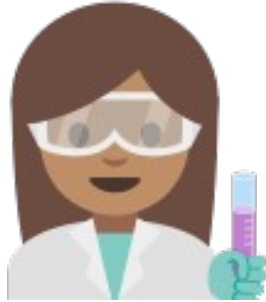


# Workflow

- Organization of cells
- Execution Order
- Insertion, Removal and Re-arrangement of Cells

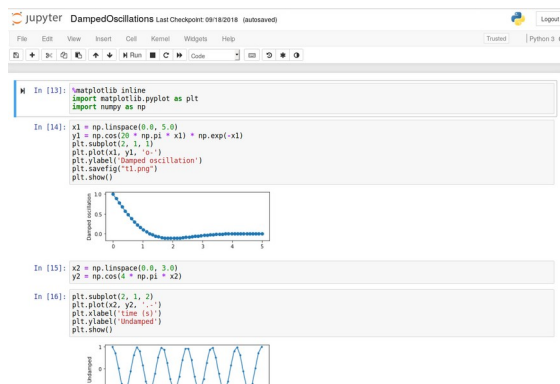


# Computational Experiments



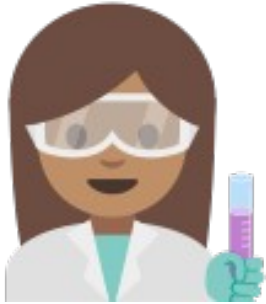
# Computational Experiments

## ➤ Repeatability

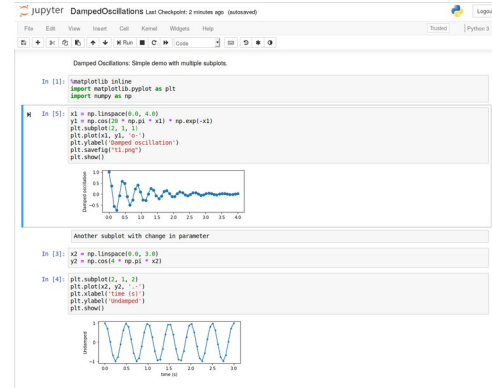
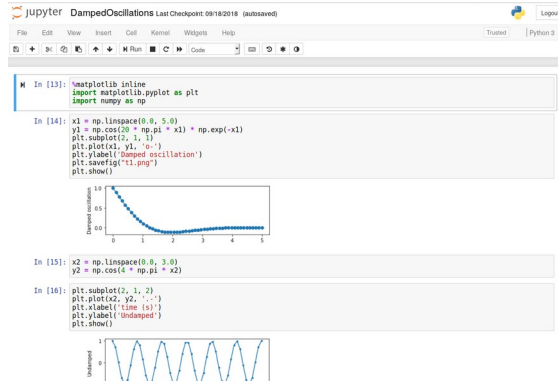


# Computational Experiments

## ➤ Repeatability



## ➤ Reproducibility



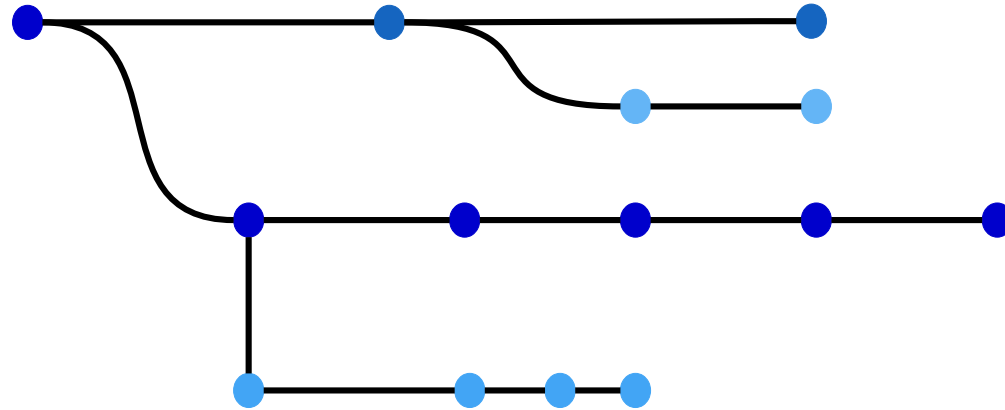
# Reproducibility

A scientific experiment is said to be **reproducible** if the experiment can be performed to get the same or similar (close-by) results by making variations in the original experiment.\*

\*A provenance-based semantic approach to support understandability, reproducibility, and reuse of scientific experiments. [Samuel, 2019]

# Provenance

*The source or origin of an object; its history*



# Computational Reproducibility

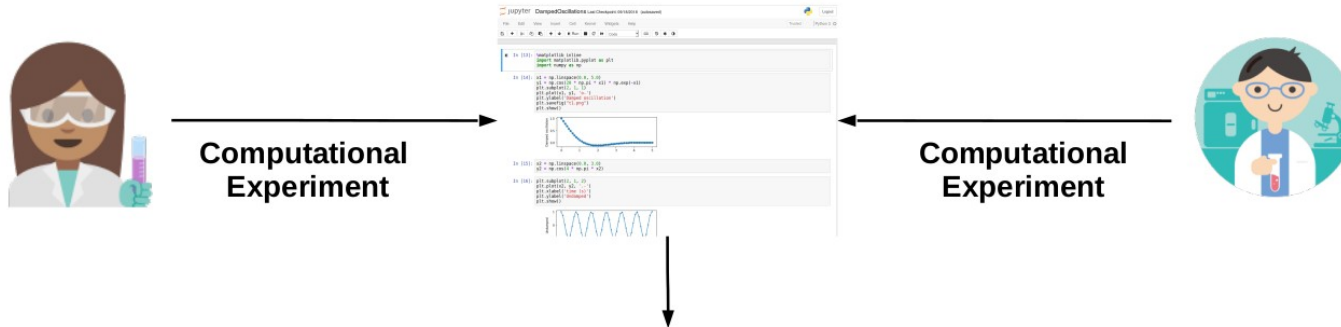


The key components for the end-to-end provenance management for computational reproducibility

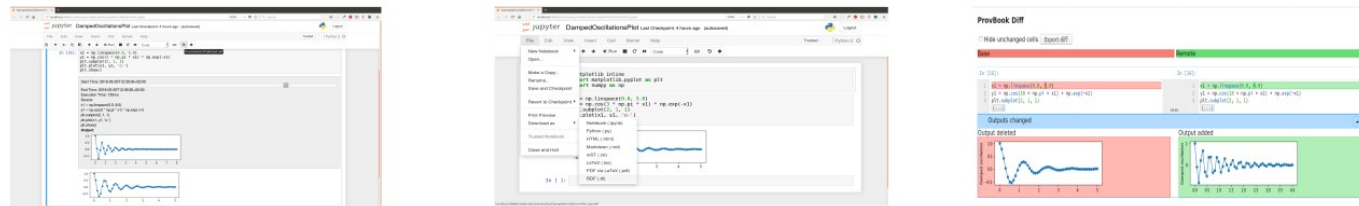
# Computational Reproducibility

- Provenance support is limited [Rule et al., 2018, Pimentel et al., 2019]
  - Tracking provenance when the cells are over-written and re-run
  - Track how exactly a final result has been achieved
  - Track of the experiments that have been attempted
- “Record all intermediate results in a standardized format”
  - One of the ten simple rules for computational reproducible research [Sandve et al., 2013]

# ProvBook: Provenance Info in Notebook



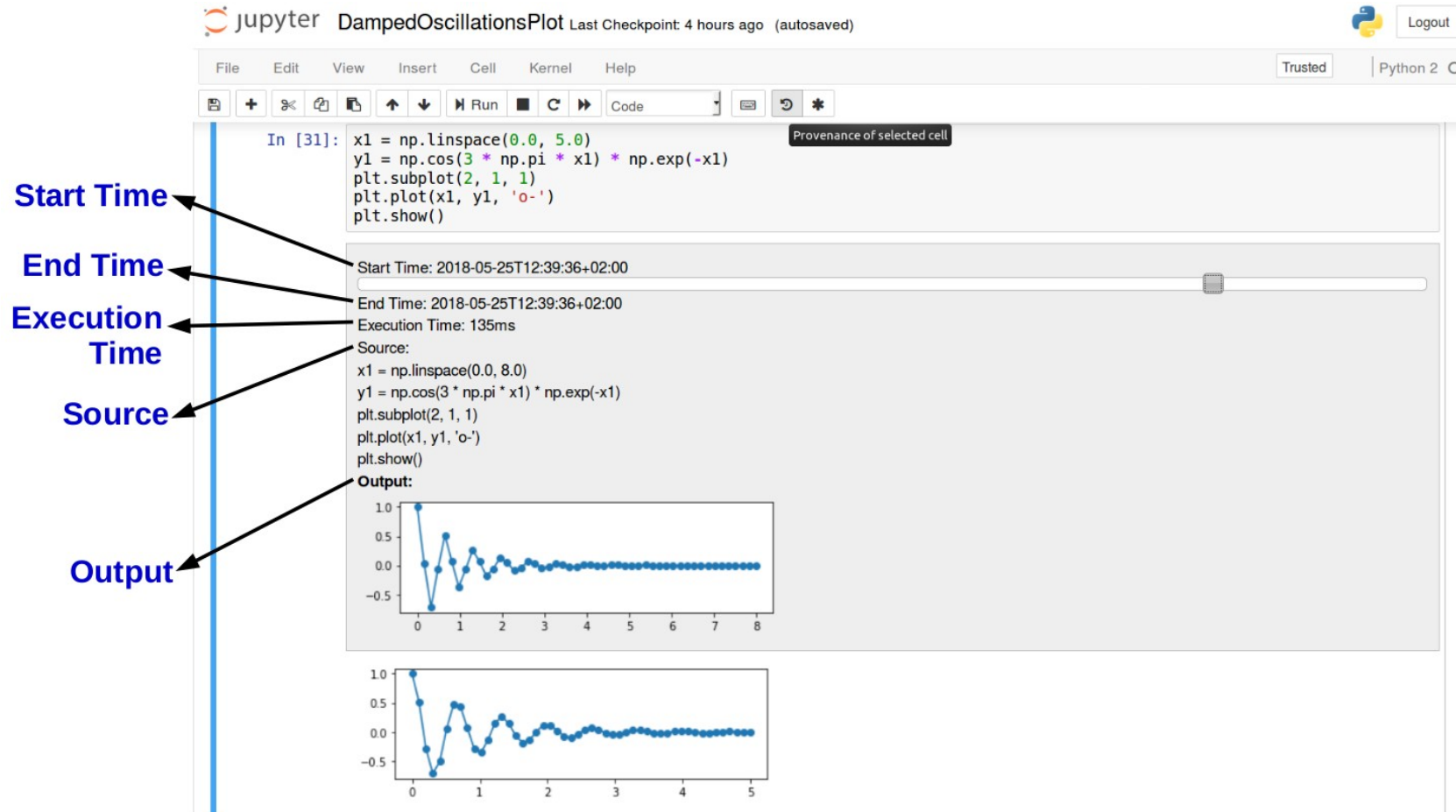
## ProvBook: Provenance of the Notebook



**Computational Reproducibility**



# ProvBook: Capture & Visualization



# ProvBook: Difference

- A provenance difference module to compare the different executions of a notebook
- Comparison of the input and the output
- Starting time to differentiate between two executions
- Extends the nbtime library from the Project Jupyter



## ProvBook Diff

☐ Hide unchanged cells Export diff

Base

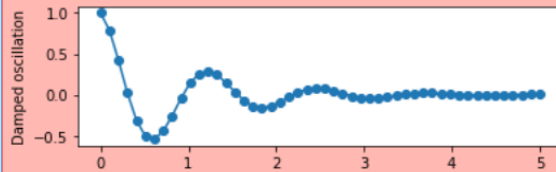
Remote

In [16]:

```
1 x1 = np.linspace(0.0, 5.0)
2 y1 = np.cos(18 * np.pi * x1) * np.exp(-x1)
3 plt.subplot(2, 1, 1)
(...)
```

Outputs changed

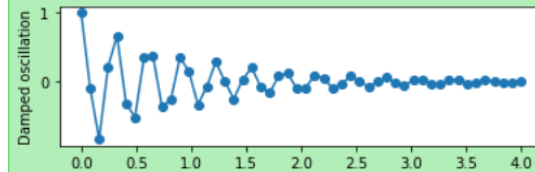
Output deleted



In [16]:

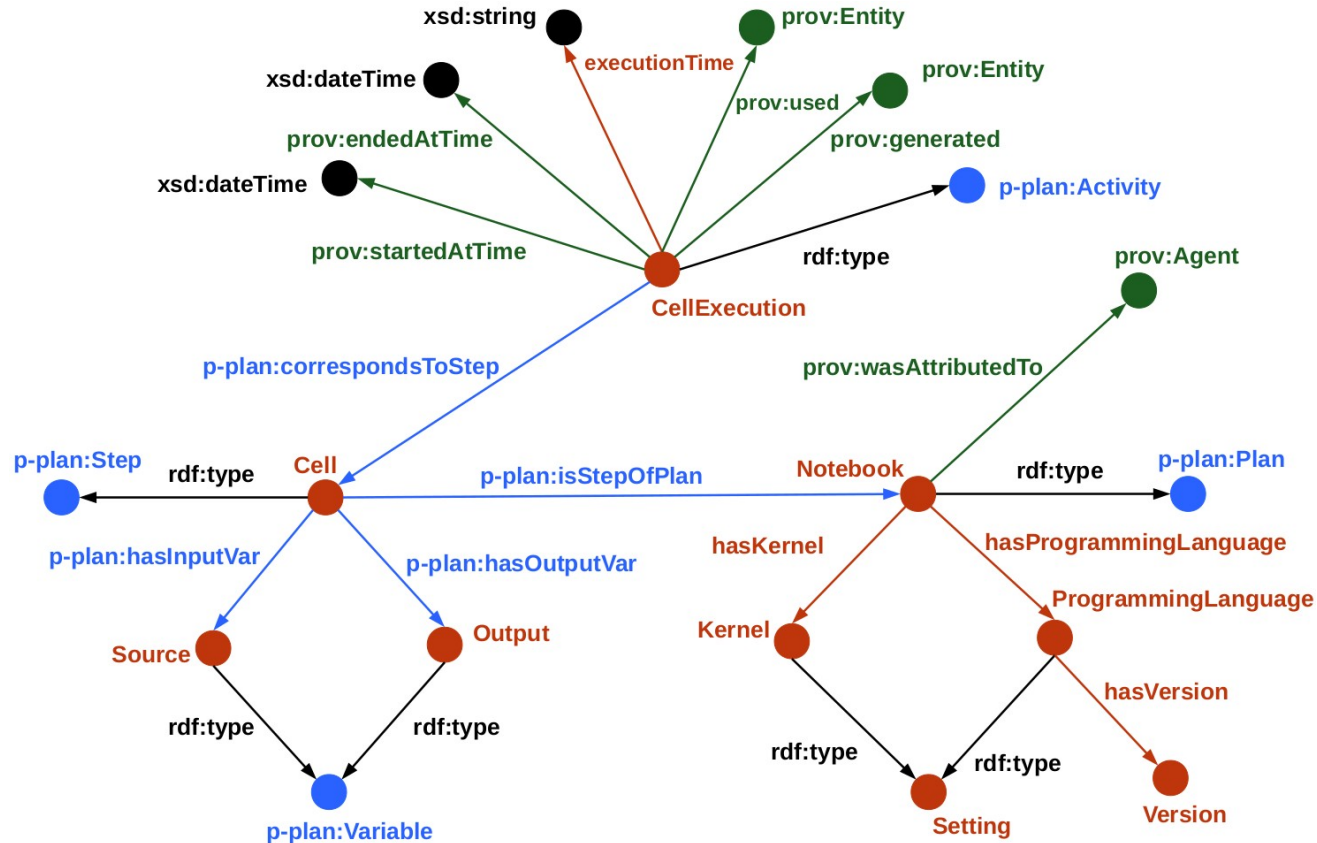
```
1 x1 = np.linspace(0.0, 4.0)
2 y1 = np.cos(18 * np.pi * x1) * np.exp(-x1)
3 plt.subplot(2, 1, 1)
(...)
```

Output added



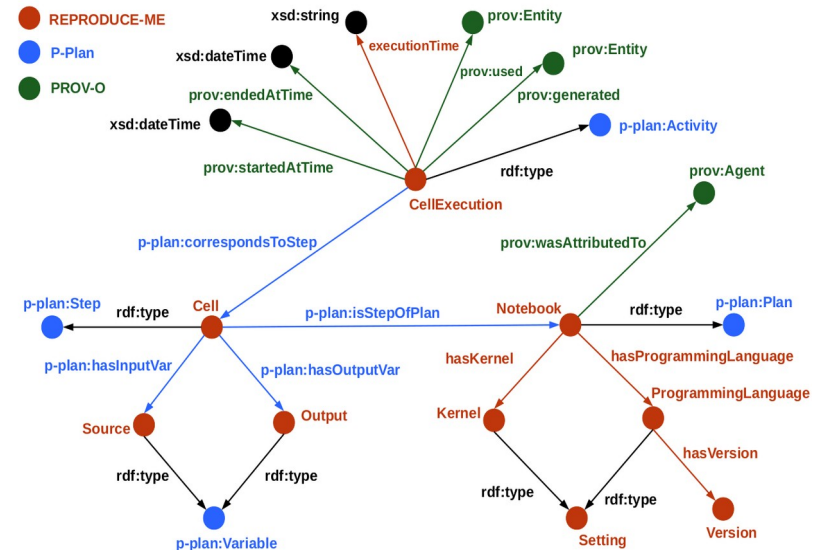
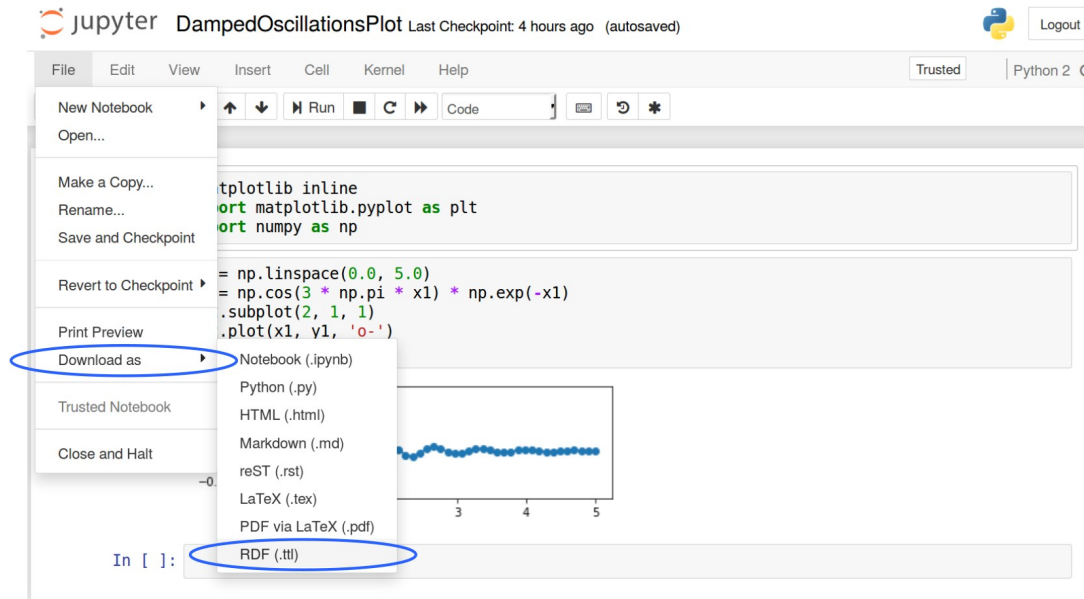
# ProvBook: Semantic Representation

An ontology is a formal, explicit specification of a shared conceptualization -[Studer et al., 1998]



# ProvBook: Semantic Representation

- Jupyter Notebook and its provenance described using the REPRODUCE-ME ontology
- **New extension:** Export in RDF from the user interface or command line



Millions of repositories in platforms like GitHub

Millions of computational notebooks in GitHub

**Are all these notebooks in repositories reproducible?**

**A single button to reproduce them?**

**REPRODUCE**



# ReproduceMeGit

- A visualization tool for analyzing the reproducibility of Jupyter Notebooks.
- Goals:
  - Help repository users and owners to reproduce, directly analyze and assess the reproducibility of notebooks
  - Built on top of the work by [Pimentel et al., 2019]
  - Get information on notebooks
    - that were successfully reproducible
    - that resulted in exceptions during runs
  - Analyze the notebooks:
    - the difference in the results from the original notebooks
    - provenance history of runs

## ReproduceMeGit

GitHubURL

Reproduce

# An Overview

## Reproducibility Study

Notebooks (un-)successfully finishing the executions

Notebooks with same or different results compared to the original.

Exceptions occurred in the runs  
ImportError, ModuleNotFoundError  
FileNotFoundError, IOError, SyntaxError

Provenance History in RDF  
using REPRODUCE-ME ontology

Direct access to  
Binder and ProvBook

**ReproduceMeGit**

## Structure & Usage

Repository Overview

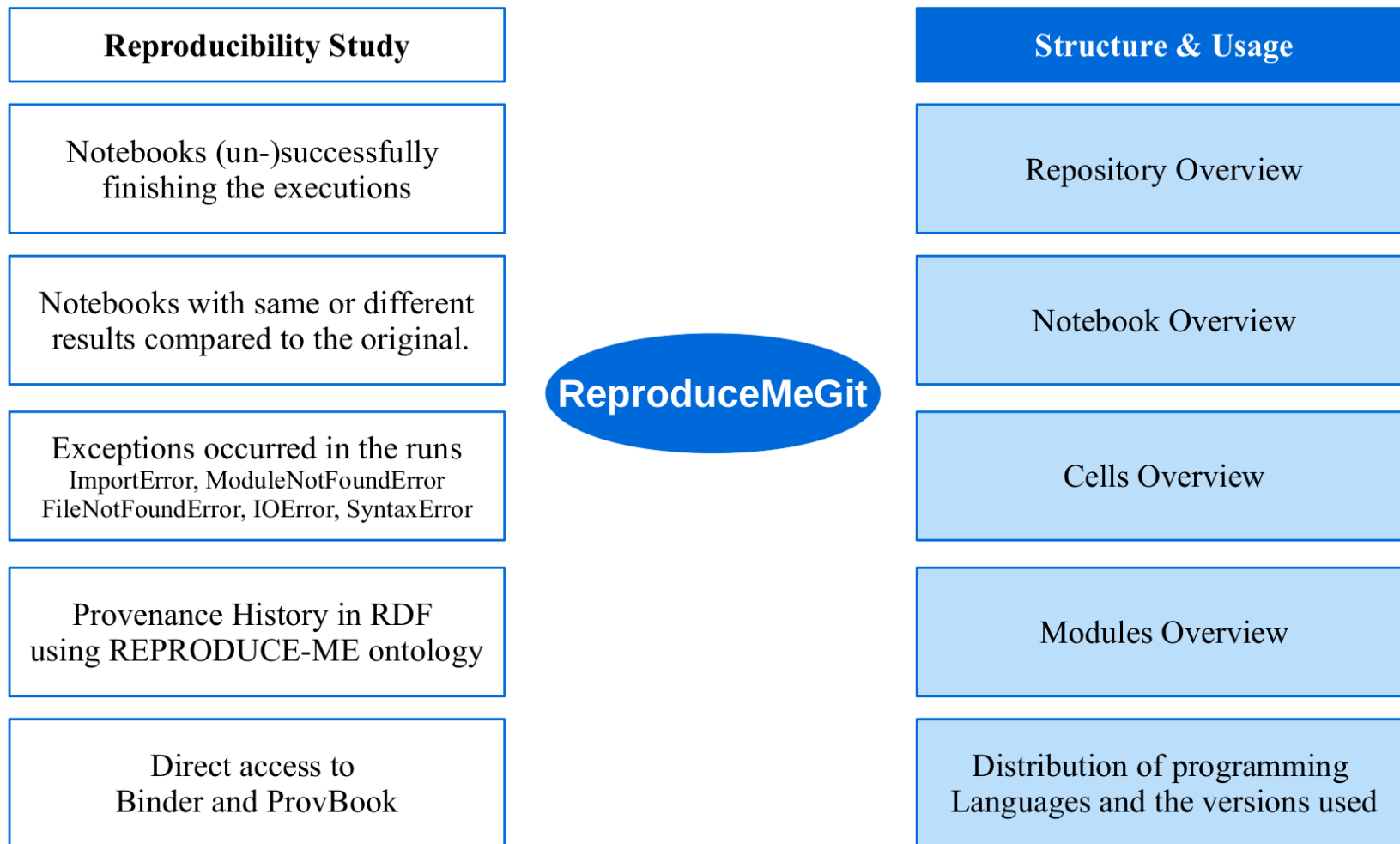
Notebook Overview

Cells Overview

Modules Overview

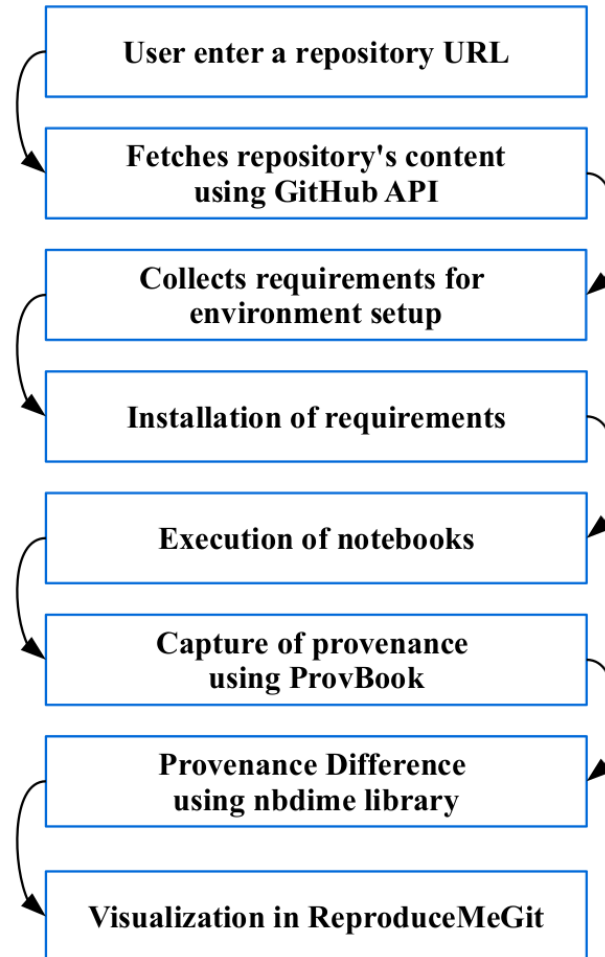
Distribution of programming  
Languages and the versions used

# An Overview





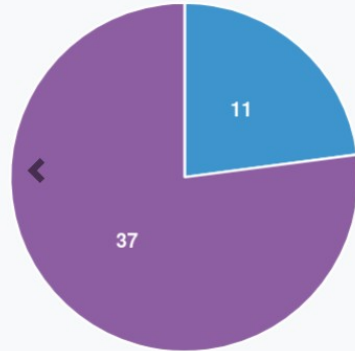
# ReproduceMeGit: Flow



# ReproduceMeGit

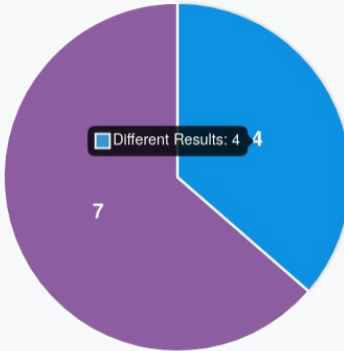
ReproduceMeGit [Home](#) [About](#)

Finished Executions Unfinished Executions



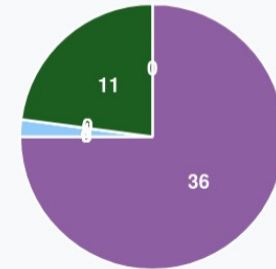
Finished vs Unfinished Executions

Different Results Same Value



Different vs Same Notebook Results

ImportError ModuleNotFoundError NameError FileNotFoundError  
IOError OSError OperationalError TypeError ValueError  
HTTPError SyntaxError AttributeError StdinNotImplementedError  
Success



Exceptions during Notebook Execution

[Repository Overview](#) [Notebook Overview](#) [Modules Overview](#) [Execution Overview](#) [ProvBook Overview](#)

#	Notebook Name	Execution Status	Execution Message	Diff On Cells	Diff Count	Execution Duration
1	MLPractical13(MinMaxScaler).ipynb	FileNotFoundError: [Errno 2] File /home/diwakar/Downloads/IITK ML Data/indians-diabetes.data.csv does not exist	Traceback (most ...	0	1	2ms
2	MLPractical21(matplotlib example).ipynb	Success	None	1,0,2,3,4,5,6	7	3ms

# Conclusion and Future Work

- Jupyter Notebooks
  - Data exploration, visualization, results
  - Reproducible research
- ProvBook, an extension of Jupyter Notebooks, supports computational reproducibility
- ReproduceMeGit: performs reproducibility study on repositories containing Jupyter Notebooks
- Tracking metadata from Jupyter Notebooks
- Reproducibility study of Jupyter Notebooks from GitHub using ReproduceMeGit

# Thanks

## Acknowledgement

- FUSION (<https://fusion.cs.uni-jena.de>)
- Carl-Zeiss Foundation

## More Information

### ProvBook Demo Video:

<https://doi.org/10.6084/m9.figshare.6401096.v2>

### ProvBook Source Code:

<https://github.com/Sheeba-Samuel/ProvBook>

### ProvBook Publication:

<http://ceur-ws.org/Vol-2180/paper-57.pdf>

### ReproduceMeGit Demo Video:

<https://doi.org/10.6084/m9.figshare.12084393.v1>

### ReproduceMeGit Source Code:

<https://github.com/fusion-jena/ReproduceMeGit>

### ReproduceMeGit Publication:

<https://arxiv.org/abs/2006.12110>



@sheebasamuel



0000-0002-7981-8504

- <https://fusion.cs.uni-jena.de/fusion/members/sheeba-samuel/>
- <https://w3id.org/reproduceme/research>

# References

- **[Lebo et al., 2013]** PROV-O: The PROV Ontology
- **[Garijo and Gil, 2012]** Augmenting PROV with plans in P-Plan: scientific processes as linked data
- **[Wilkinson et al., 2016]** The FAIR Guiding Principles for scientific data management and stewardship.
- **[Adam Rule et al.]** Exploration and explanation in computational notebooks.
- **[Pimentel et al., 2019]** A large-scale study about quality and reproducibility of jupyter notebooks
- **[Project Jupyter et al., 2018]** Binder 2.0 - Reproducible, interactive, sharable environments for science at scale.
- **[Samuel, 2019]** A provenance-based semantic approach to support understandability, reproducibility, and reuse of scientific experiments
- **[Samuel and König-Ries, 2018b]** ProvBook: Provenance-based semantic enrichment of interactive notebooks for reproducibility.
- **[Samuel and König-Ries, 2017]** REPRODUCE-ME: Ontology-Based Data Access for Reproducibility of Microscopy Experiments.
- **[Samuel and König-Ries, 2018a]** Combining P-Plan and the REPRODUCE-ME ontology to achieve semantic enrichment of scientific experiments using interactive notebooks.
- **[Samuel et al., 2018]** The Story of an Experiment: A Provenance-based Semantic Approach towards Research Reproducibility.
- **[Samuel and König-Ries, 2020]** ReproduceMeGit: A Visualization Tool for Analyzing Reproducibility of Jupyter Notebooks.