Research Data Management in Neuroscience

An introduction to Jupyter notebooks

Michael Sonntag

Department Biologie II
Ludwig-Maximilians-Universität München

Friday, 26 June, 2020

Overview of Jupyter notebook qualities

- tool to reconcile code and documentation using the webbrowser
  - like a labbook for code and plots
  - easy to view and hand over
- can be used with different scripting languages
  - Python
  - R
  - (Matlab)
- has tools built around it
  - can be used for presentations
  - can be shared and read online e.g. using nbviewer
  - can be shared, run and modified online e.g. using binder

Find a short introduction at

- https://nbviewer.jupyter.org/github/jupyter/notebook/blob/master/docs/source/examples/Notebook/Notebook%20Basics.ipynb (https://nbviewer.jupyter.org/github/jupyter/notebook/blob/master/docs/source/examples/Notebook/Notebook%20Basics.ipynb)

Detour 1: Package managers and virtual environments

A package manager

- keeps track of installed software packages
- provides a list of available software packages
- enables the installation of software packages from available off-site repositories
- many different ones exist for OS system packages or programming language libraries
  - e.g. `apt`, `yum`, `pip`, `homebrew`, `chocolatey`

A virtual environment

- is a tool that keeps software packages separate from the operating system
- can be used to install different versions of the same software
- can be used to document and re-create and share an analysis environment

(Ana)Conda is a cross platform (Linux, macOS, Windows) package manager and a virtual environment.

Conda installation and usage resources

Installation files and instructions

- https://docs.conda.io/en/latest/miniconda.html (https://docs.conda.io/en/latest/miniconda.html)

20 minute introduction to conda

- https://conda.io/projects/conda/en/latest/user-guide/getting-started.html (https://conda.io/projects/conda/en/latest/user-guide/getting-started.html)

Conda commands cheat sheet

- https://docs.conda.io/projects/conda/en/latest/_downloads/843d9e0198f2a193a3484886fa28163c/conda-cheatsheet.pdf (https://docs.conda.io/projects/conda/en/latest/_downloads/843d9e0198f2a193a3484886fa28163c/conda-cheatsheet.pdf)

Working with conda - essential commands I

- open a terminal
- create a new conda virtual environment

```
conda create -n [env_name] python=3.8
```

- activate the environment; Operating System (OS) software packages are no longer available

```
conda activate [env_name]
```

- install software packages using the `conda` package manager; will be installed into the active environment only

```
conda install [library]
```

- install python packages using `pip` ; when using `pip` , it will be installed into the active environment only

```
pip install [library]
```

Working with conda - essential commands II

- deactivate an active environment; the installed packages are no longer available. The OS packages are now available again.

```
conda deactivate
```

- delete an environment with all installed software packages

```
conda remove -n [env_name] --all
```

Conda pitfalls and issues

Installing into the base conda installation

- Do not use `conda activate` .
- Immediately do `conda deactivate` .
- Otherwise the installation of conda will be affected and can become unusable.

Conda terminal in Windows

- Start menu; search for and open "Anaconda Prompt"

Conda pitfalls and issues

Conda and the Windows directory path issue

Python/Conda known issue on Windows: whitespaces in directory paths

- https://github.com/conda/conda/issues/8725 (https://github.com/conda/conda/issues/8725)
- https://docs.anaconda.com/anaconda/user-guide/faq/ (https://docs.anaconda.com/anaconda/user-guide/faq/)

```
"
In what folder should I install Anaconda on Windows?

We recommend installing ...conda into a directory that
- contains only 7-bit ASCII characters and no spaces ...
- Do not install into paths that contain spaces such
    as C:\Program Files or include Unicode characters ...
"
```

Anaconda claims that they are compliant, but that third party packages that will be installed might not be compliant.

Finally: Jupyter notebooks

"Project Jupyter exists to develop open-source software, open-standards, and services for interactive computing across dozens of programming languages."

https://jupyter.org/

- all Jupyter notebooks require a Python installation
- Python is the default scripting language in Jupyter notebooks

Oh no, Detour 2: Introduction to Python

Not by this course

Here are three links to lessons tailored for scientists; can each be done in 1/2 - 1 day:

- https://swcarpentry.github.io/python-novice-inflammation (https://swcarpentry.github.io/python-novice-inflammation)
- https://swcarpentry.github.io/python-novice-gapminder (https://swcarpentry.github.io/python-novice-gapminder)
- https://datacarpentry.org/python-ecology-lesson (https://datacarpentry.org/python-ecology-lesson)

They include

- Python essentials: very basics, functions, exceptions, debugging
- Data loading, handling and plotting
- Libraries for Data handling

Installation example of Jupyter in a conda environment I

- open a (conda) terminal
- create the python conda environment

```
conda create -n jnb-py38 python=3.8 -y
```

- activate the environment

```
conda activate jnb-py38
```

- install jupyter into the active conda environment

```
pip install jupyter
```

- navigate to a working directory on the operating system

```
cd [path/to/working/directory]
```

- open jupyter

```
jupyter notebook
```

Installation example of Jupyter in a conda environment II

- a notebook server will start and will switch automatically to the browser
- select `New -> Notebook: Python3`
- opens another tab with an unsaved notebook
- save the notebook under a name
- to close a notebook, switch to the terminal
- press `ctrl+C` (`⌘+C` on macOS) to close the notebook server
- you can directly start an existing notebook from the command line

```
jupyter notebook [file_name.ipynb]
```

We'll now do a quick feature glimpse on a local Jupyter notebook.

You can also check out this curated list of published notebooks to get an idea what you can actually do with it.

https://github.com/jupyter/jupyter/wiki/A-gallery-of-interesting-Jupyter-Notebooks (https://github.com/jupyter/jupyter/wiki/A-gallery-of-interesting-Jupyter-Notebooks)

Two interesting examples from the neuroscientific field:

- https://nbviewer.jupyter.org/github/arokem/teach_optimization/blob/master/optimization.ipynb (https://nbviewer.jupyter.org/github/arokem/teach_optimization/blob/master/optimization.ipynb)
- https://nbviewer.jupyter.org/github/wtadler/cue-combination-with-neurons/blob/master/neural_cue_combination.ipynb (https://nbviewer.jupyter.org/github/wtadler/cue-combination-with-neurons/blob/master/neural_cue_combination.ipynb)

Using Jupyter notebook

Check the reference for FAQs and details

- https://jupyter-notebook.readthedocs.io (https://jupyter-notebook.readthedocs.io)

Find a quick tutorial about the basic features here

- https://realpython.com/jupyter-notebook-introduction (https://realpython.com/jupyter-notebook-introduction)

Jupyter cells for notes use a specific flavor of markdown (see Lecture05)

- https://jupyter-notebook.readthedocs.io/en/stable/examples/Notebook/Working%20With%20Markdown%20Cells.html (https://jupyter-notebook.readthedocs.io/en/stable/examples/Notebook/Working%20With%20Markdown%20Cells.html)

Make your life easier

Use keyboard shortcuts

The running notebook provides a full list of available shortcuts via the notebook menu:

```
`Help` -> `Keyboard Shortcuts`
```

A nice rundown also comparing across OS platforms can be found here

- https://towardsdatascience.com/jypyter-notebook-shortcuts-bf0101a98330 (https://towardsdatascience.com/jypyter-notebook-shortcuts-bf0101a98330)

Use magic methods

Jupyter specific magic methods (subset of Python magic)

```
"IPython has a system of commands we call 'magics' that provide
 a mini command language ... and is extensible by the user with
 new commands."
```

- https://nbviewer.jupyter.org/github/ipython/ipython/blob/6.x/examples/IPython%20Kernel/Cell%20Magics.ipynb (https://nbviewer.jupyter.org/github/ipython/ipython/blob/6.x/examples/IPython%20Kernel/Cell%20Magics.ipynb)

Python magic methods

- https://ipython.readthedocs.io/en/stable/interactive/magics.html (https://ipython.readthedocs.io/en/stable/interactive/magics.html)

This is the full set of magic methods; Jupyter magic methods do not fully support all Python magic methods.

Use widgets with your code

Interactive plots in Jupyter notebooks using the `%matplotlib` magic method

- https://www.mikulskibartosz.name/interactive-plots-in-jupyter-notebook (https://www.mikulskibartosz.name/interactive-plots-in-jupyter-notebook)
- plots then enable zoom, save, ...

Interactive elements - sliders, dropdowns, textfields

- https://ipywidgets.readthedocs.io/en/latest/index.html (https://ipywidgets.readthedocs.io/en/latest/index.html)

Use Jupyter notebooks as presentations

- In a running notebook select from the menu bar:

```
`View` -> `Cell toolbar` -> `Slideshow`
```

- You can now select how each cell should appear in a presentation.
- After you have saved your notebook and closed Jupyter, restart Jupyter as a presentation tool:

```
jupyter nbconvert [file_name.ipynb] --to slides --post serve
```

- When running the jupyter notebook, it will create a `[file_name].slides.html` file which can be used in presentations as well.
- Use the following command to create a static html page.

```
jupyter nbconvert --to html [file_name.ipynb]
```

- This HTML file can be converted to a PDF and used as a handout.

Additional features when publishing notebooks

Uploading Jupyter notebooks to public git repositories gives access to additional tools

- can be published on any online git repository e.g. github, gitlab, gin ...
- online browsing of published Jupyter notebooks content via `NBViewer`
  - https://nbviewer.jupyter.org (https://nbviewer.jupyter.org)
  - e.g. https://nbviewer.jupyter.org/github/arokem/teach_optimization/blob/master/optimization.ipynb (https://nbviewer.jupyter.org/github/arokem/teach_optimization/blob/master/optimization.ipynb)
- online editing and sharing of published Jupyter notebooks via `Binder`
- Keep in mind: all of these features work best, if requirements and notebooks are in the root of a repository

Binder

Binder is a free service that enables to run and work on a published Jupyter notebook on a remote service machine.

- minimal set up required
- run full Jupyter notebooks remote - no local installation required

The full documentation in how to properly use Binder can be found at

- https://mybinder.readthedocs.io/en/latest (https://mybinder.readthedocs.io/en/latest)

You can find minimal examples how to set up a repository for use with Binder for Python and R at

- https://github.com/binder-examples (https://github.com/binder-examples)

We will do a very quick introduction into how to set up and use Python and R Binder notebooks.

Empty Binder set up for Python Jupyter notebooks I

- Prepare an empty, public git repository
- To start a Python Jupyter notebook via Binder you need to provide two files at the root of the repository.
  - `runtime.txt` ... this file contains the Python version that will be used for any Notebook started by this repository. It contains only one entry:

    ```
    python-3.8
    ```

  - `requirements.txt` ... this file contains the Python packages that will be installed when the container starts. In our example we will install the following python packages:

    ```
    numpy
    matplotlib
    nixio==1.5.0b4
    ```

- Commit and upload these files to the public git repository
- Check the following lecture repo as an example
  - https://gin.g-node.org/RDMcourse2020/demo-lecture-06 (https://gin.g-node.org/RDMcourse2020/demo-lecture-06)

Empty Binder set up for Python Jupyter notebooks II

- Go to https://mybinder.org (https://mybinder.org)
- Select `Git repository` and paste the URL of the repository e.g. https://gin.g-node.org/RDMcourse2020/demo-lecture-06 (https://gin.g-node.org/RDMcourse2020/demo-lecture-06)
- Select `Launch` ; it can take some time until the environment is ready for use.
- You can now create a new notebook vie the menu `New -> Notebook: Python3` .
- A new tab will open and you can save it under a new name - this will still be on the remote machine.
- Use the `Download` menu button fetch the notebook to your machine.
- You can now work on this notebook as you would locally.

Python Binder with an existing notebook

- Set up a git repository with environment and Python dependencies
- Upload an existing Jupyter notebook to this repo and any data files to be used in this notebook
- Use https://mybinder.org (https://mybinder.org) to set up the repository information as before
- Now also provide the file name of the uploaded Jupyter notebook and select `Launch`
- Binder will now launch the notebook directly

Notes on Binder usage

Binder is a free service

- building a `container` (running environment) might take up to 20min the first time around.
- takes longer the more dependencies are defined.
- built containers are kept for a while - the next time the Binder container is used, the start up will take less time.

Binder is a cloud service; all files are remote

- upload required dependencies and files to the git repository
- upload required files to the running cloud service
- save changes to your notebook in the cloud service AND download the notebook to your local machine
- Binder will time out when there is inactivity - save OFTEN

Notes on Binder usage

https://mybinder.org (https://mybinder.org) can provide a permanent link to public Binder compatible git repositories

- you can run your notebook from anywhere
- collaborators can easily run and try out your notebook as well

You can have multiple Jupyter notebooks in the same repository

- all can have their own permanent links
- all will share the same environment and dependencies

Jupyter and R

Full example of an R Jupyter notebook set up via conda

- Required the additional installation of an `R kernel`
- The `R kernel` enables the notebook to "understand" R syntax

```
conda create -n r-jnb-py38 python=3.8 -y
conda activate r-jnb-py38
conda install -c conda-forge jupyterlab -y
conda install -c r r-base -y
conda install -c r r-irkernel -y

jupyter notebook
```

- From the menu select `New -> Notebook: R`

Additional resources to set up Jupyter with R

Jupyter and R - setup and use

- https://www.datacamp.com/community/blog/jupyter-notebook-r (https://www.datacamp.com/community/blog/jupyter-notebook-r)
- https://datatofish.com/r-jupyter-notebook/ (https://datatofish.com/r-jupyter-notebook/)
- https://plotly.com/r/using-r-in-jupyter-notebooks/ (https://plotly.com/r/using-r-in-jupyter-notebooks/)

R maintains its own channel on conda and provides all libraries for conda environments

- https://anaconda.org/r (https://anaconda.org/r)

Useful R libraries available via conda

- `r-base`
  - core R installation
- `r-tidyverse`
  - scientific data package collection
  - https://tidyverse.tidyverse.org (https://tidyverse.tidyverse.org)
  - it contains
    - ggplot2, for data visualisation.
    - dplyr, for data manipulation.
    - tidyr, for data tidying.
    - readr, for data import.
    - ...
- Installation via conda

```
conda install -c r r-base
conda install -c r r-tidyverse
```

R Jupyter notebook with Binder set up example I

- We again need a public git repository
- This time we need to provide the YAML file `environment.yml` at the root of the repository.
- It contains all `conda` packages that need to be installed to run the `R` Jupyter notebook in Binder.

```
channels:
  - r
dependencies:
  - r-base
  - r-tidyverse
```

- Upload the file to the public git repository

R Jupyter notebook with Binder set up example II

- Go to https://mybinder.org/ (https://mybinder.org/)
- Select `Git repository` and paste the URL of the repository e.g. https://gin.g-node.org/RDMcourse2020/demo-lecture-06 (https://gin.g-node.org/RDMcourse2020/demo-lecture-06)
- Select `launch` ; it will now take a bit until the environment is created and ready for you to use.
- You can now create a new notebook vie the menu `New -> Notebook: R` . Note, that we also still have the Python dependencies in the same repository. That is why we could also start a Python3 notebook as well.
- A new tab will open with the Jupyter notebook. Note the `R` logo which denotes that we are working in an R Jupyter notebook.

Jupyter and Matlab

Matlab - live editor and Matlab online

- https://www.mathworks.com/products/matlab/live-editor.html (https://www.mathworks.com/products/matlab/live-editor.html)
- https://www.mathworks.com/products/matlab-online.html (https://www.mathworks.com/products/matlab-online.html)

Matlab in Jupyter notebooks:

- https://anneurai.net/2015/11/12/matlab-based-ipython-notebooks (https://anneurai.net/2015/11/12/matlab-based-ipython-notebooks)

Matlab kernels for Jupyter notebooks

- https://github.com/Calysto/matlab_kernel (https://github.com/Calysto/matlab_kernel)
- https://github.com/imatlab/imatlab (https://github.com/imatlab/imatlab)

Publishing Jupyter notebooks

Course notes on Jupyter notebooks and good practice:

- https://reproducible-science-curriculum.github.io/publication-RR-Jupyter/ (https://reproducible-science-curriculum.github.io/publication-RR-Jupyter/)

???