

Papermill

Jupyter Notebooks run in Batch

Jupyter Notebooks

Jupyter Notebooks:

- Nice Interface for FAST implementations (Mathematica Notebook-like)
- Data Exploration
- Visualization

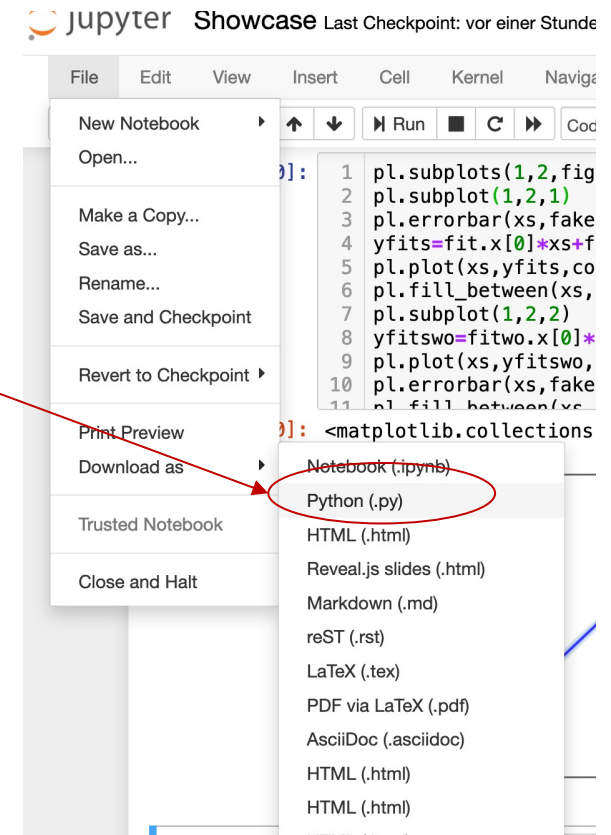
Caveats:

- Nonlinear execution
- Not easily scriptable
 - input parameters?



Jupyter Notebooks

- Usual workflow(s):
 - Type A:
 - Develop in Jupyter notebook
 - Export notebook to script
 - Run script in batch system
 - Type B:
 - Explore data
 - Fix some aspect of data analysis
 - Run script in batch system with fixed parameters




Papermill

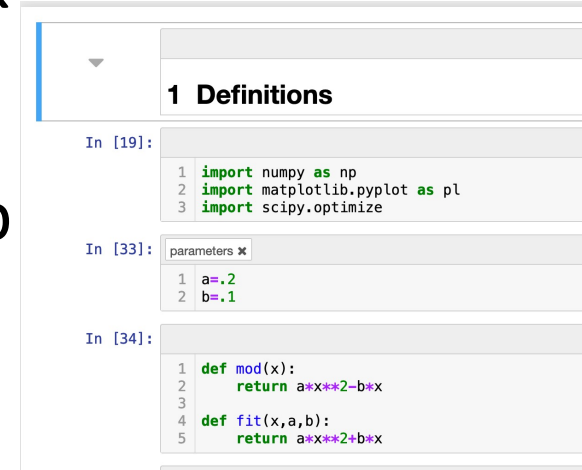
- Would be nice to
 - Run notebooks with arguments
 - Create output-notebooks with mixed output (strings, plots, etc.)
 - Interactively debug failing notebooks
 - Run this in batch



<https://papermill.readthedocs.io/en/latest/#>

Papermill

- Idea of  **papermill**
 - Have a „template“ notebook
 - Declare parameters as input
 - Run the template notebook with input parameters
 - Save the output in a new notebook
- Example:
 - Template notebook `template.ipynb`
 - Two parameters: `a` and `b`
 - Run in command line

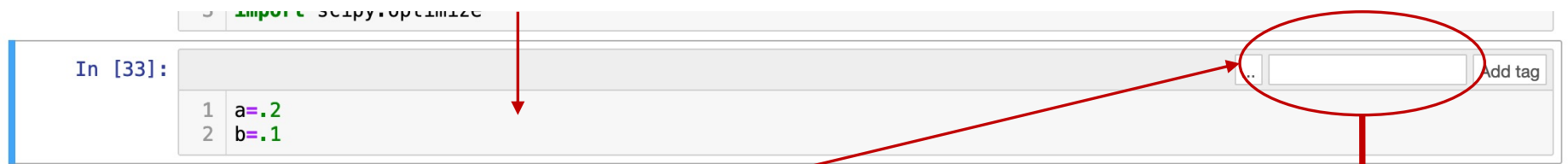


```
In [19]:  
1 import numpy as np  
2 import matplotlib.pyplot as plt  
3 import scipy.optimize  
  
In [33]: parameters  
1 a=.2  
2 b=.1  
  
In [34]:  
1 def mod(x):  
2     return a*x**2-b*x  
3  
4 def fit(x,a,b):  
5     return a*x**2+b*x
```

```
(venv) [djukanov@login21 papermill]$ papermill template.ipynb run1.ipynb -p a 2.2 -p b 2.55  
Input Notebook:  template.ipynb  
Output Notebook: run1.ipynb  
Executing: 100%
```

Papermill

- How to get input
 - Take ordinary cell



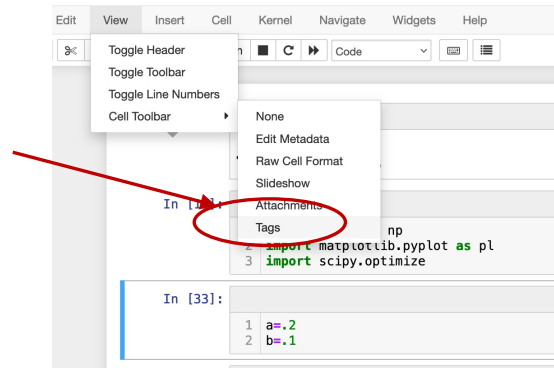
A screenshot of a Jupyter Notebook cell. The cell contains two lines of code: `1 a=.2` and `2 b=.1`. The cell is labeled "In [33]:". To the right of the code area, there is a small input field with a red circle around it, containing an ellipsis "...". To the right of this field is a button labeled "Add tag". A red arrow points from the top of the cell down to the input field.

- Set tag of the cell as: parameters



A screenshot of a Jupyter Notebook cell, identical to the one above. The cell contains the same code: `1 a=.2` and `2 b=.1`. The cell is labeled "In [33]:". The input field to the right now contains the text "parameters" and is circled in red. The "Add tag" button is still visible to the right.

- If **Add tag** is not visible toggle it



- That's all!

Papermill



```
1 Definitions
In [19]:
1 import numpy as np
2 import matplotlib.pyplot as plt
3 import scipy.optimize

In [33]: parameters x
1 a=.2
2 b=.1

In [34]:
1 def mod(x):
2     return a*x**2-b*x
3
4 def fit(x,a,b):
5     return a*x**2+b*x
```

```
1 Definitions
In [1]:
1 import numpy as np
2 import matplotlib.pyplot as plt
3 import scipy.optimize

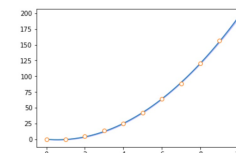
In [2]: parameters x
1 a=.2
2 b=.1

In [3]: injected-parameters x
1 # Parameters
2 a = 2.2
3 b = 2.55
4

In [4]:
1 def mod(x):
2     return a*x**2-b*x
3
4 def fit(x,a,b):
5     return a*x**2+b*x
```

```
In [12]:
1 fvals=fit(np.linspace(0,10),pval)
2 plt.plot(np.linspace(0,10),fvals)
3 plt.fill_between(np.linspace(0,10),fvals+err,fvals-err,color='b',alpha=.2)
4 plt.errorbar(datx,daty,yerr=2,fmt='o',mfc='w',capsize=2)

Out[12]: <ErrorbarContainer object of 3 artists>
```



- Example Output:
 - Output notebook run1.ipynb
 - Two parameters: a and b
 - Set in command line
 - Some result based on injected parameters

Papermill

- Command line has reduced set of options
- Run from within python

```
#!/usr/bin/env python

import papermill as pm

pm.execute_notebook('./template.ipynb', './run2.ipynb', parameters=dict(a=2, b=25))
~
```

- Parameters as dict:
Can pass more complicated things (e.g. arrays)

```
def run_D200():
    for bins in [2]:
        for binloops in [1]:
            pol=3
            for po in ['pol0_', '']:
                ff=ret_runs('D200')
                if po == 'pol0_':
                    flo=16
                    fup=28
                else:
                    flo=16
                    fup=31
            try:
                pm.execute_notebook('./Sigma_term_papermill.ipynb', './output/Sigma_D200_bin_'+str(bins)+'_po_'+po+'.ipynb', parameters=
                    dict(ensn='D200', ensr='D200r000', datadir='./', files_get=ff, flower=flo, fupper=fup,
                        bins=bins, smd=1, po=po, pol=3, bins_loops=binloops,
                        outfname_sigmapinff='gs_eff_D200_bin_'+str(bins)+'_bin_loop_'+str(binloops)+po+'_sigmapin_ff.h5',
                        outfname_scalarff='gs_eff_D200_bin_'+str(bins)+'_bin_loop_'+str(binloops)+po+'_scalar_ff.h5',
                        outdir='./output/', rmcfgs=[[757]]))
            except:
                print(traceback.format_exc())
                continue
```

Papermill

- Run in Slurm just as any other script

```
#!/bin/bash
#SBATCH -J papermill           # Job name
#SBATCH -p himster2_exp       # Queue name 'short' or 'long' on Mogon I
                              # 'smp' on Mogon II
#SBATCH -n 1
#SBATCH --ntasks-per-node=32  # Total number of tasks, here explicitly 1
#SBATCH -t 05:30:00          # Run time (hh:mm:ss) - 0.5 hours
#SBATCH --hint=nomultithread
#SBATCH -A m2_him_exp        # Specify allocation to charge against

export HDF5_USE_FILE_LOCKING='FALSE'

module load mpi/OpenMPI/4.0.3-GCC-9.3.0
source /home/djukanov/venv/bin/activate

export PYTHONPATH=$PYTHONPATH:/home/djukanov/LaQAPack/src/python/LaQAPack

./run_papermill.py -e $1
```

- In case of an error Cell is output



The screenshot shows a Jupyter Notebook interface. The browser address bar indicates the URL is localhost:8888/notebooks/git/sigma_term/analysis/output/Sigma_C101_bin_2_po_ipynb. The notebook title is "Sigma_C101_bin_2_po_ Last Checkpoint: 25.11.2021 (autosaved)". The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Navigate, Widgets, Help) and a toolbar with icons for file operations and execution. On the left, a "Contents" sidebar shows a tree view with "3 Runs" expanded, including "3.1.1 Connected" and "3.1.2 Disconnected". The main area displays a code cell with the following content:

```
In [ ]:
1 %html
2 <span style="color:red; font-family:Helvetica Neue, Helvetica, Arial, sans-serif; font-size:2em;">An Except
```

Below the code cell, a red error message is displayed: "An Exception was encountered at 'In [38]'".

Papermill

- In case of an error Cell is output



The screenshot shows a Jupyter Notebook interface. The browser address bar indicates the URL is localhost:8888/notebooks/git/sigma_term/analysis/output/Sigma_C101_bin_2_po_ipynb. The notebook title is 'Sigma_C101_bin_2_po_ Last Checkpoint: 25.11.2021 (autosaved)'. The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Navigate, Widgets, Help) and a toolbar with icons for file operations and execution. On the left, a 'Contents' sidebar shows a tree view with folders for 'Comments', 'Programs', and 'Runs'. The main area contains a code cell with the following content:

```
In [ ]:
```

```
1 %%html
2 <span style="color:red; font-family:Helvetica Neue, Helvetica, Arial, sans-serif; font-size:2em;">An Except
```

Below the code cell, the output is displayed in red text: "An Exception was encountered at 'In [38]'".

- Run the cell up to the error
- Debug interactively
- Include changes in template
- Rerun

END