

Modelling Advisory Group (MAG)

Hydrological Uncertainty Workshop

Global Sensitivity Analysis using the SAFE toolbox

23 September 2025

Francesca Pianosi
University of Bristol



Access to software, papers, example: <https://safetoolbox.github.io/>



University of
BRISTOL



Science and
Technology
Facilities Council

Pianosi is funded under the DAFNI Centre of Excellence for Resilient Infrastructure Analysis within the UKRI Building a Secure and Resilient World program, with grant number ST/Y003713/1

How to run a Jupyter Notebooks from browser (no need to install anything on your computer!)

Overview:

1. Download all the required files (Jupyter Notebook + data files + python code files) to your computer from:

<https://github.com/FrancescaPianosi/GSAtraining/>

2. Upload and run the Jupyter Notebook in your browser through the page:

<https://colab.research.google.com> (Google Colab - if you have a Google account)

<https://safetoolbox.github.io/demos/lab/index.html> (Jupyterlite - if you do not have a Google account)

We will use the Jupyter Notebook called *GR6J_GSA_Notebook.ipynb*

but if the interactive visualization is too slow or gets stuck, use *GR6J_GSA_Notebook_static.ipynb* instead
(this is more likely to happen if you are using the Jupyterlite)

Detailed instructions to download the required files

1. Download all the required files to your computer

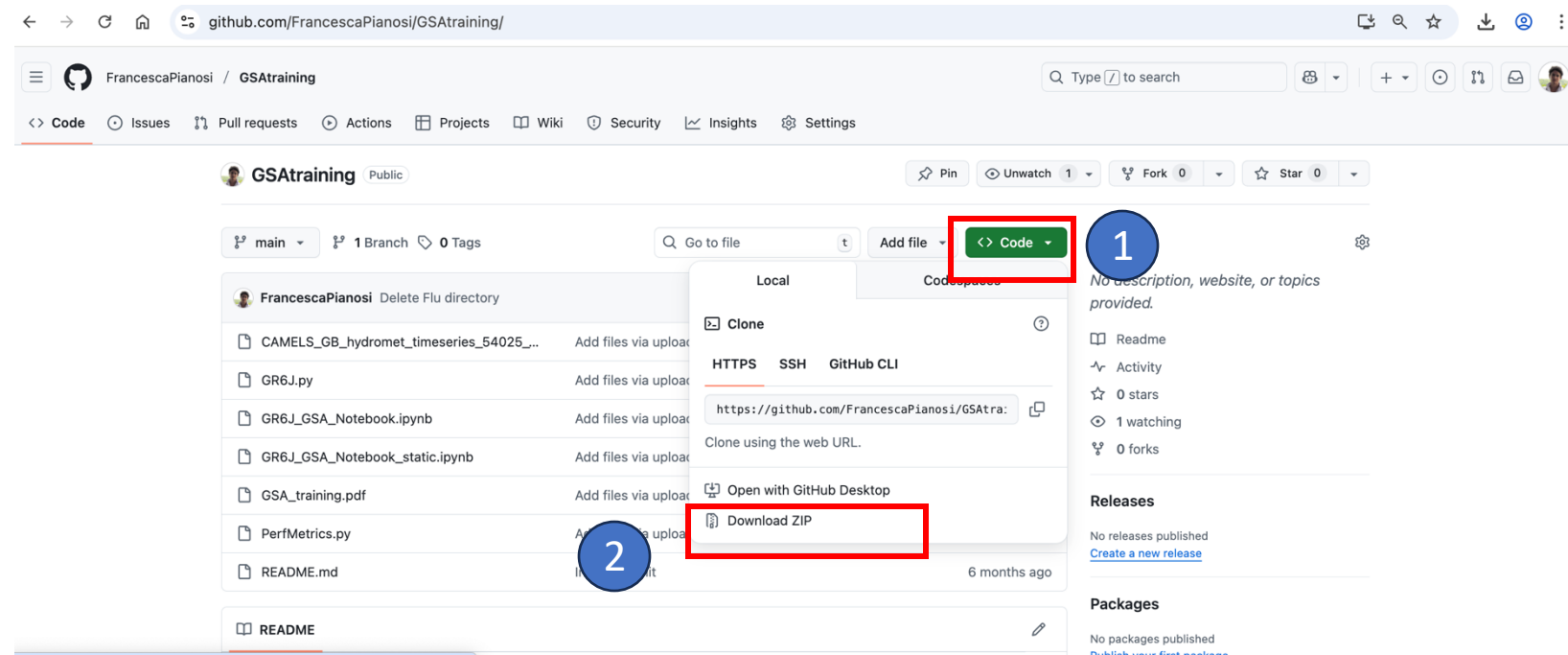
Go to: <https://github.com/FrancescaPianosi/GSAtraining/>

Click on

< > Code

Download ZIP

Unzip the folder on your computer



Detailed instructions if you are using Google Colab

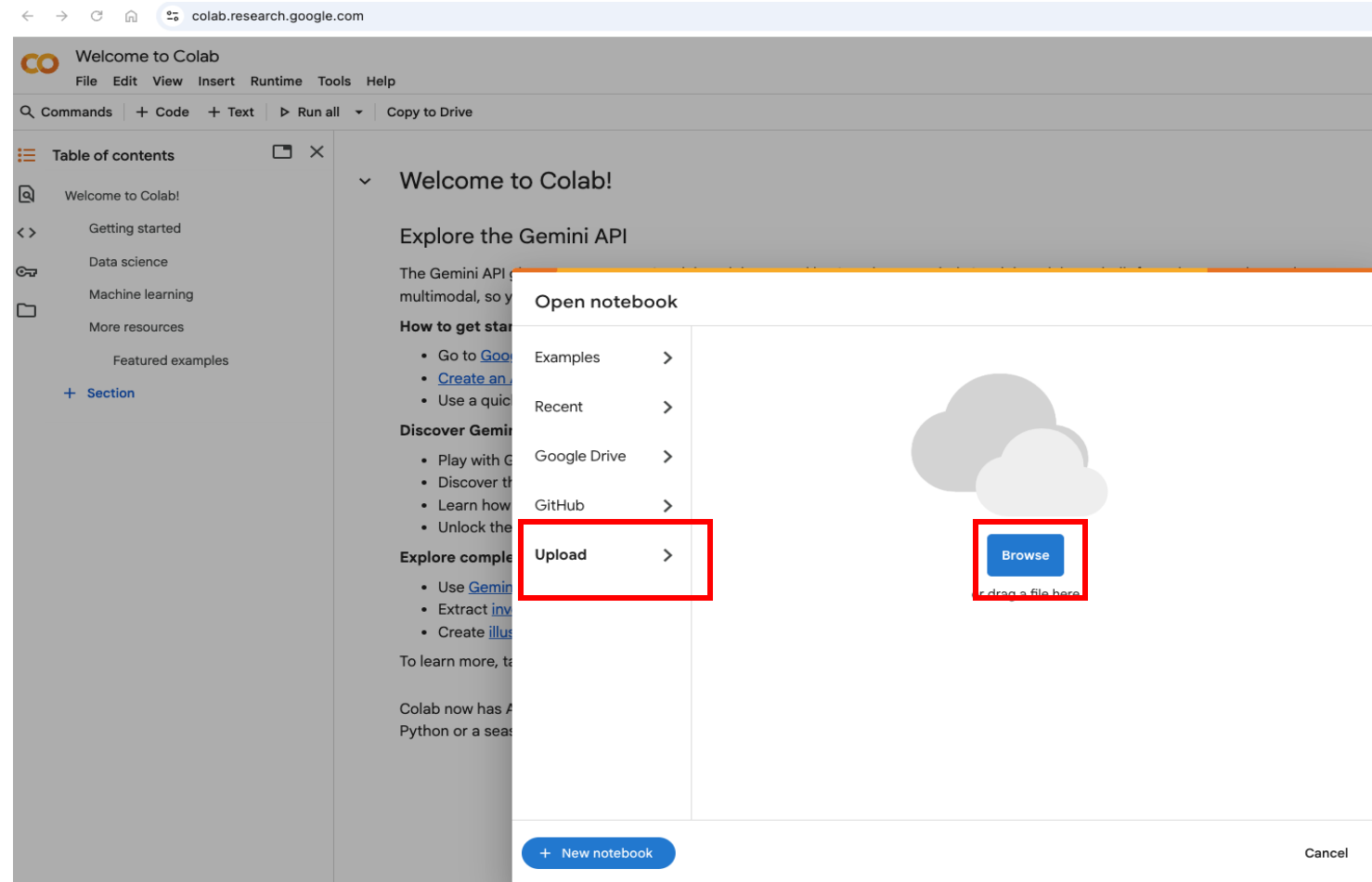
2. Upload the Jupyter Notebook in your browser using Google Colab

Go to: <https://colab.research.google.com>

A pop-up window should automatically open

Choose “Upload”

Find the *GR6J_GSA_Notebook.ipynb* file on your computer

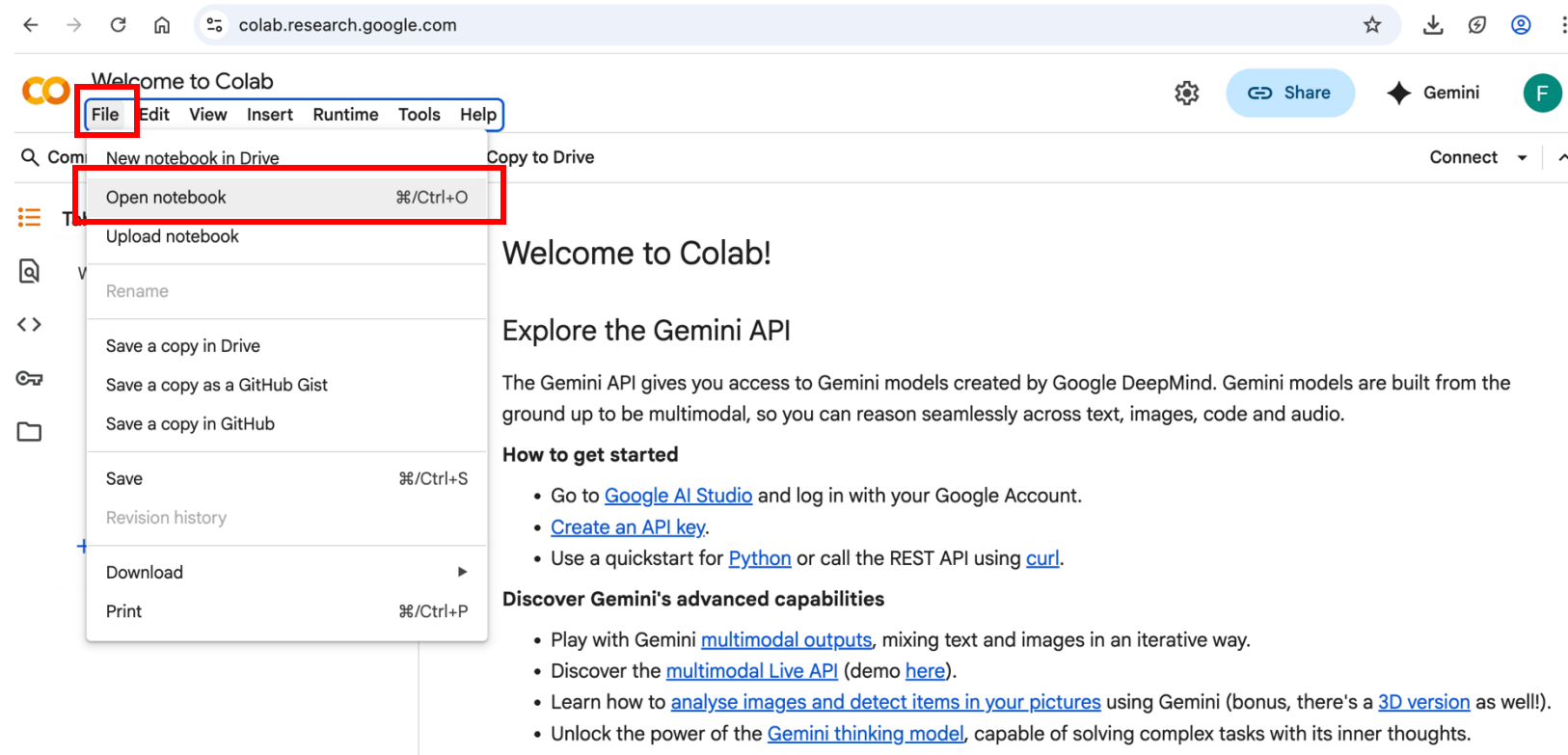


2. Upload the Jupyter Notebook in your browser using Google Colab - *Troubleshooting*

If the upload pop-up window does not open automatically open, go to:

File

Open notebook



3. Upload other required files (.csv data file and two .py code files)

Copy of GR6J_GSA_Notebook.ipynb

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all

Files

2

1

It takes a few seconds to show the Files page.

3

4

Open

PhD > Projects > Francesca Pianosi > GSA_training

Name	Status	Date modified	Type	Size
CAMELS_GB_hydromet_timeseries_54025...	✓	9/18/2025 3:02 PM	Microsoft Excel C...	1,046 KB
GR6J	✓	9/18/2025 3:02 PM	JetBrains PyChar...	11 KB
GR6J_GSA_Notebook	✓	9/18/2025 3:02 PM	Jupyter Source File	289 KB
Notes	↻	9/18/2025 3:35 PM	Microsoft PowerP...	903 KB
PerfMetrics	✓	9/18/2025 3:02 PM	JetBrains PyChar...	5 KB

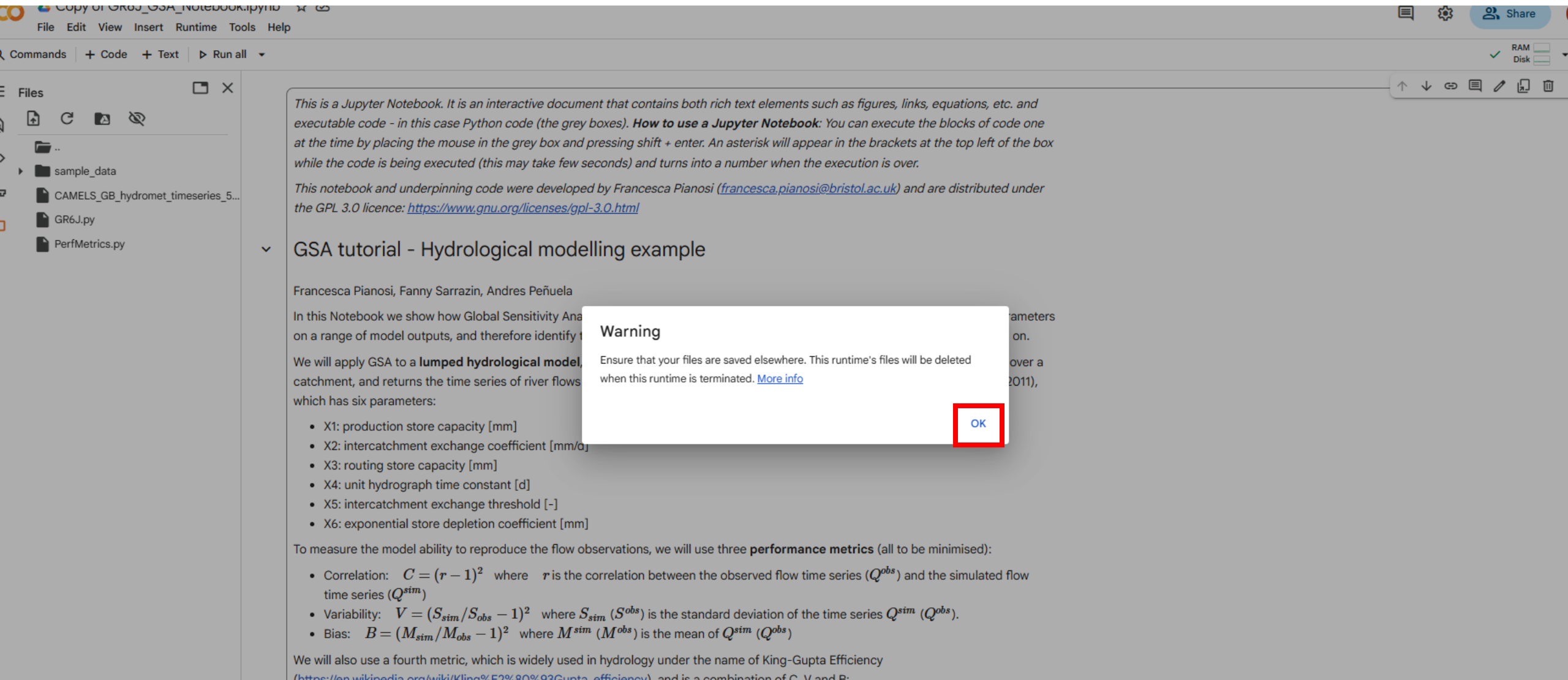
File name: "CAMELS_GB_hydromet_timeseries_54025_19701001-20150930" "GR6J" "PerfMetrics"

All Files

Open Cancel

- X3: routing store capacity [mm]
- X4: unit hydrograph time constant [d]
- X5: intercatchment exchange threshold [-]

4. Ignore the warning message



The screenshot shows a Jupyter Notebook interface. On the left is a file explorer with a folder named 'sample_data' and files 'CAMELS_GB_hydromet_timeseries_5...', 'GR6J.py', and 'PerfMetrics.py'. The main area displays the notebook content, which includes a title 'GSA tutorial - Hydrological modelling example' and introductory text about the notebook's purpose and authors. A warning dialog box is overlaid on the notebook content, with the title 'Warning' and the message: 'Ensure that your files are saved elsewhere. This runtime's files will be deleted when this runtime is terminated. [More info](#)'. The 'OK' button in the dialog box is highlighted with a red square. The notebook content also lists six parameters (X1 to X6) and three performance metrics (Correlation, Variability, Bias) used for model evaluation.

This is a Jupyter Notebook. It is an interactive document that contains both rich text elements such as figures, links, equations, etc. and executable code - in this case Python code (the grey boxes). **How to use a Jupyter Notebook:** You can execute the blocks of code one at the time by placing the mouse in the grey box and pressing shift + enter. An asterisk will appear in the brackets at the top left of the box while the code is being executed (this may take few seconds) and turns into a number when the execution is over.

This notebook and underpinning code were developed by Francesca Pianosi (francesca.pianosi@bristol.ac.uk) and are distributed under the GPL 3.0 licence: <https://www.gnu.org/licenses/gpl-3.0.html>

✓ GSA tutorial - Hydrological modelling example

Francesca Pianosi, Fanny Sarrazin, Andres Peñuela

In this Notebook we show how Global Sensitivity Analysis (GSA) can be applied to a hydrological model, by varying the parameters on a range of model outputs, and therefore identifying the most influential parameters.

We will apply GSA to a **lumped hydrological model**, which simulates the hydrology of a catchment, and returns the time series of river flows over a period of time (e.g. 2011), which has six parameters:

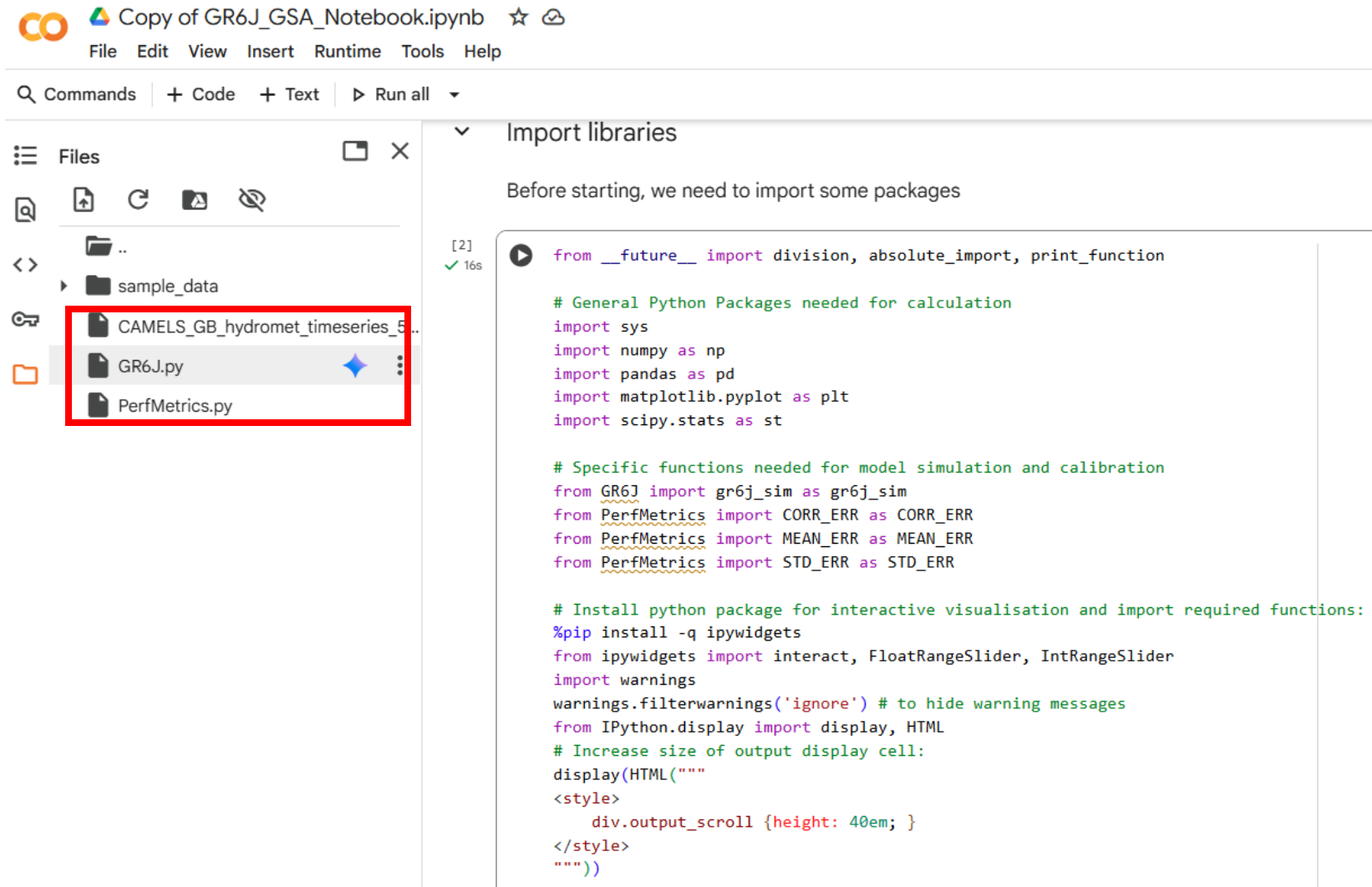
- X1: production store capacity [mm]
- X2: intercatchment exchange coefficient [mm/d]
- X3: routing store capacity [mm]
- X4: unit hydrograph time constant [d]
- X5: intercatchment exchange threshold [-]
- X6: exponential store depletion coefficient [mm]

To measure the model ability to reproduce the flow observations, we will use three **performance metrics** (all to be minimised):

- Correlation: $C = (r - 1)^2$ where r is the correlation between the observed flow time series (Q^{obs}) and the simulated flow time series (Q^{sim})
- Variability: $V = (S_{sim}/S_{obs} - 1)^2$ where S_{sim} (S_{obs}) is the standard deviation of the time series Q^{sim} (Q^{obs}).
- Bias: $B = (M_{sim}/M_{obs} - 1)^2$ where M_{sim} (M_{obs}) is the mean of Q^{sim} (Q^{obs})

We will also use a fourth metric, which is widely used in hydrology under the name of King-Gupta Efficiency (https://en.wikipedia.org/wiki/King%E2%80%93Gupta_efficiency), and is a combination of C, V and B:

5. Check that all the required files have been uploaded



The screenshot displays the JupyterLab interface for a notebook titled "Copy of GR6J_GSA_Notebook.ipynb". The top menu bar includes "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". Below the menu, there are tabs for "Commands", "+ Code", "+ Text", and a "Run all" button.

On the left, the "Files" panel shows the directory structure. A red rectangle highlights the files "GR6J.py" and "PerfMetrics.py" in the current directory, indicating they have been successfully uploaded. Other files visible include "CAMELS_GB_hydromet_timeseries_5.." and a "sample_data" folder.

On the right, the "Import libraries" section contains the text: "Before starting, we need to import some packages". Below this, a code cell [2] is shown with the following Python code:

```
[2] ✓ 16s
from __future__ import division, absolute_import, print_function

# General Python Packages needed for calculation
import sys
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import scipy.stats as st

# Specific functions needed for model simulation and calibration
from GR6J import gr6j_sim as gr6j_sim
from PerfMetrics import CORR_ERR as CORR_ERR
from PerfMetrics import MEAN_ERR as MEAN_ERR
from PerfMetrics import STD_ERR as STD_ERR

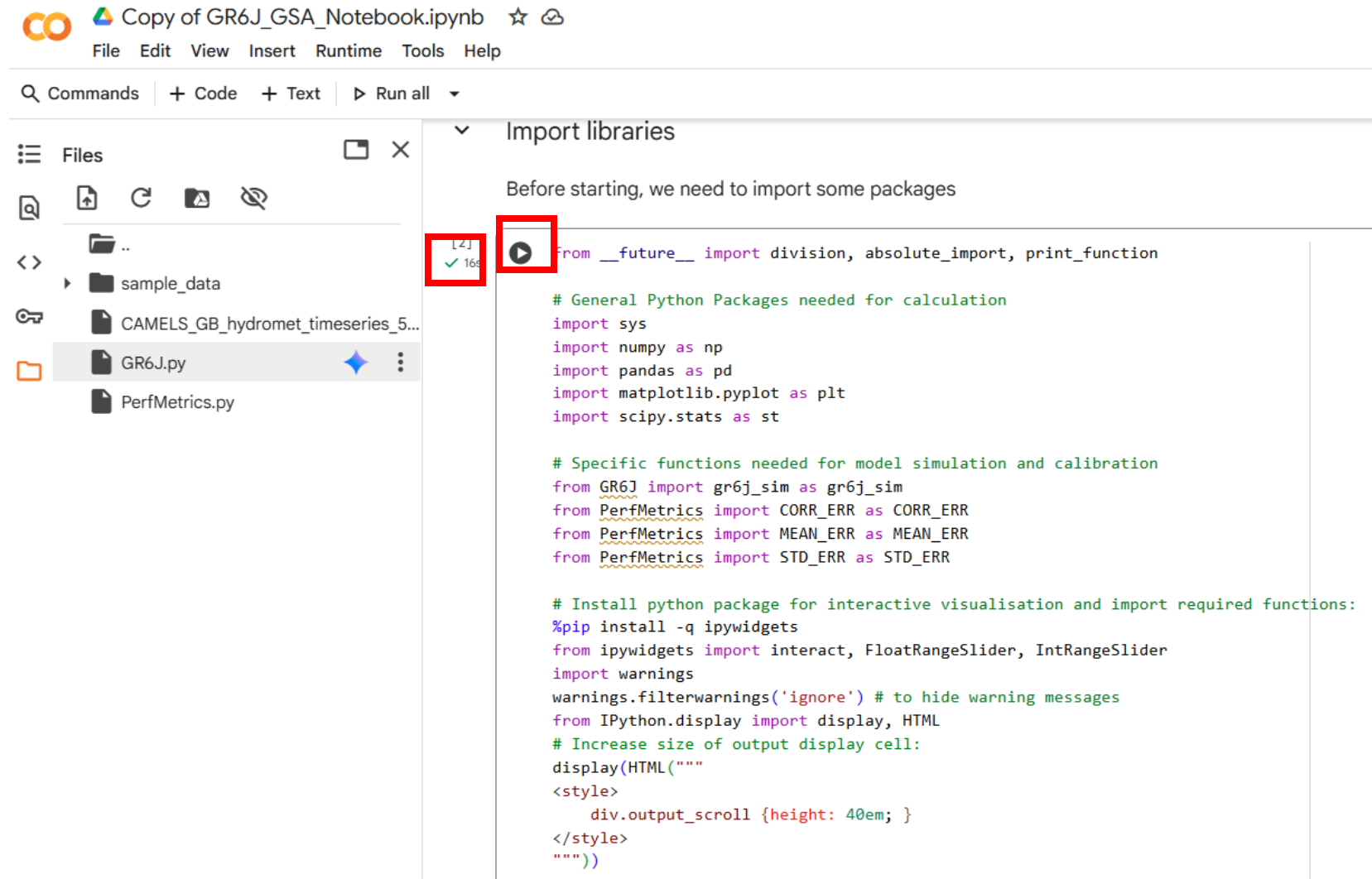
# Install python package for interactive visualisation and import required functions:
%pip install -q ipywidgets
from ipywidgets import interact, FloatRangeSlider, IntRangeSlider
import warnings
warnings.filterwarnings('ignore') # to hide warning messages
from IPython.display import display, HTML
# Increase size of output display cell:
display(HTML("""
<style>
    div.output_scroll {height: 40em; }
</style>
"""))
```

6. Run the code

The Jupyter Notebook includes cells of executable code and cells of text with explanations of what is going on

The cells of code can be executed one at the time by clicking on the “run” icon.

If the code has been executed correctly, a green tick will appear. If the execution produces any plot or numerical results, they will appear under the cell



The screenshot displays a Jupyter Notebook titled "Copy of GR6J_GSA_Notebook.ipynb". The interface includes a top menu bar with options like File, Edit, View, Insert, Runtime, Tools, and Help. Below the menu is a toolbar with icons for running code, saving, and other actions. The left sidebar shows a file explorer with a list of files: sample_data, CAMELS_GB_hydromet_timeseries_5..., GR6J.py (selected), and PerfMetrics.py. The main area shows a code cell with a green tick icon and a play button icon. The code cell contains the following Python code:

```
[2] ✓ 16s ▶ from __future__ import division, absolute_import, print_function

# General Python Packages needed for calculation
import sys
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import scipy.stats as st

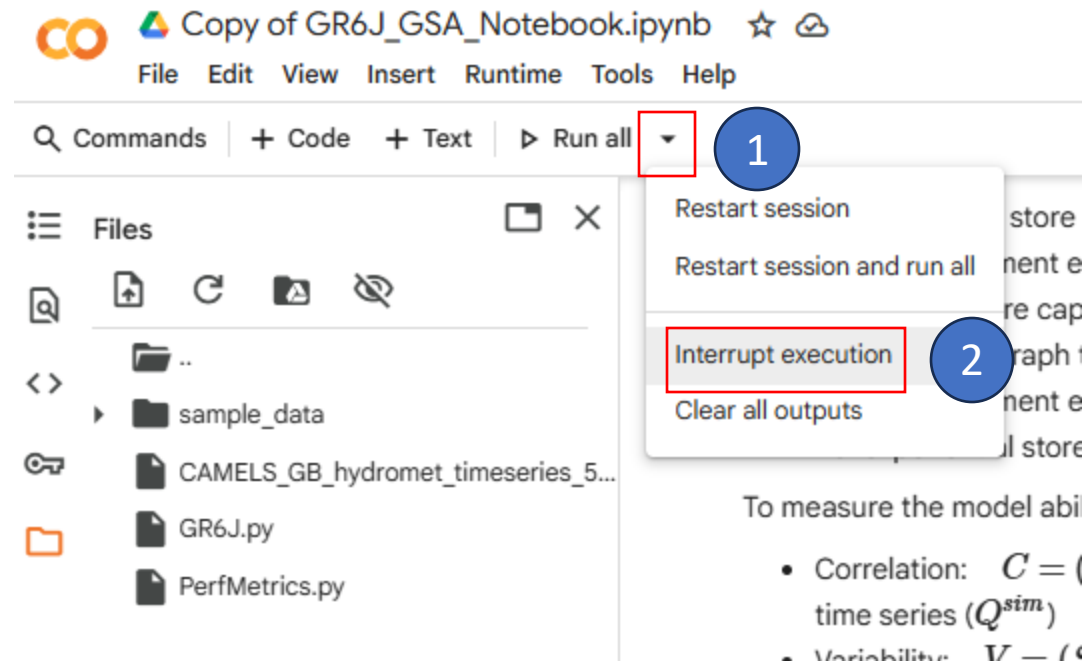
# Specific functions needed for model simulation and calibration
from GR6J import gr6j_sim as gr6j_sim
from PerfMetrics import CORR_ERR as CORR_ERR
from PerfMetrics import MEAN_ERR as MEAN_ERR
from PerfMetrics import STD_ERR as STD_ERR

# Install python package for interactive visualisation and import required functions:
%pip install -q ipywidgets
from ipywidgets import interact, FloatRangeSlider, IntRangeSlider
import warnings
warnings.filterwarnings('ignore') # to hide warning messages
from IPython.display import display, HTML
# Increase size of output display cell:
display(HTML("""
<style>
    div.output_scroll {height: 40em; }
</style>
"""))
```

6. Run the code - *Troubleshooting*

Normally all commands should run in less than a minute

If it takes longer than that, the system may be stuck and you may need to interrupt the execution and repeat



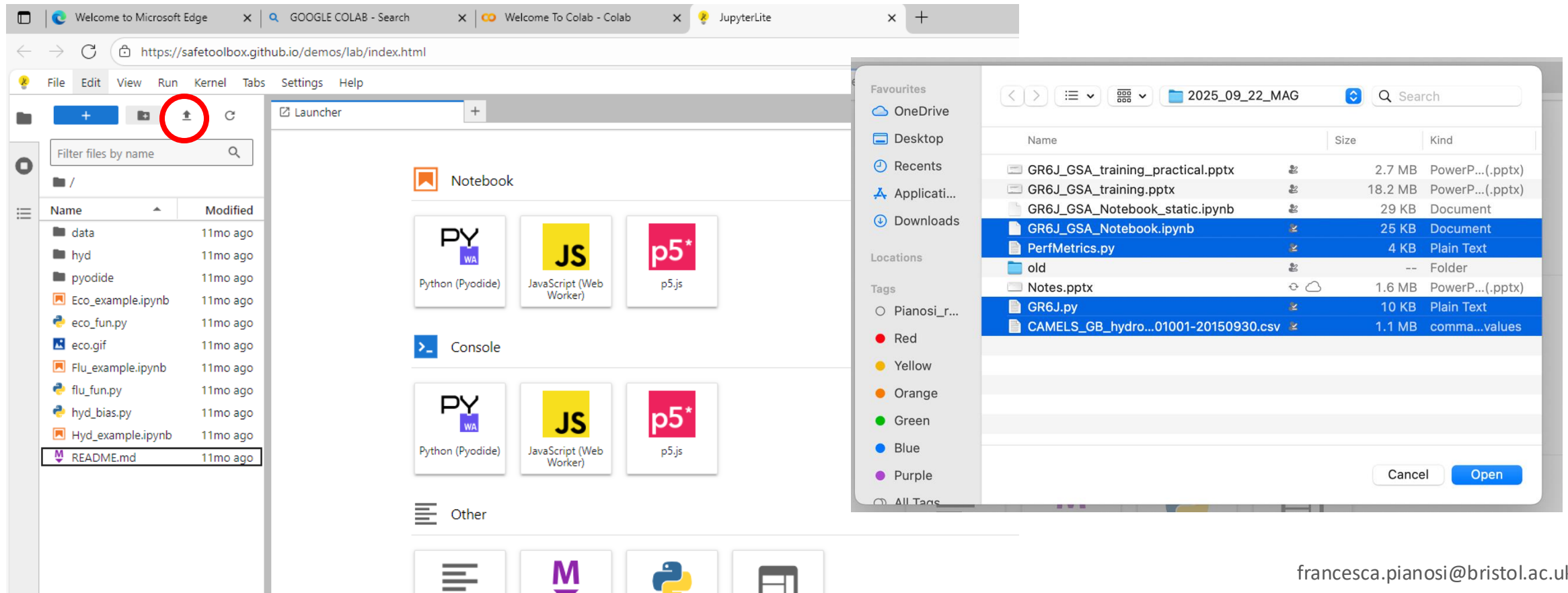
Instructions if you are using JupyterLite

2. Upload the Jupyter Notebook in your browser using Google Colab

Go to: <https://safetoolbox.github.io/demos/lab/index.html>

Click on the Upload Files icon on the top left

Find all the files needed on your computer (*GR6J_GSA_Notebook.ipynb* + *GR6J.py* and *PerfMetrics.py* + *.csv* datafile)



The screenshot shows the Google Colab interface in a Microsoft Edge browser. The 'File' menu is open, and the 'Upload Files' icon (a square with an upward arrow) is circled in red. A file explorer window is open, showing the contents of the '2025_09_22_MAG' folder. The files listed are:

Name	Size	Kind
GR6J_GSA_training_practical.pptx	2.7 MB	PowerP...(.pptx)
GR6J_GSA_training.pptx	18.2 MB	PowerP...(.pptx)
GR6J_GSA_Notebook_static.ipynb	29 KB	Document
GR6J_GSA_Notebook.ipynb	25 KB	Document
PerfMetrics.py	4 KB	Plain Text
old	--	Folder
Notes.pptx	1.6 MB	PowerP...(.pptx)
GR6J.py	10 KB	Plain Text
CAMELS_GB_hydro...01001-20150930.csv	1.1 MB	comma...values

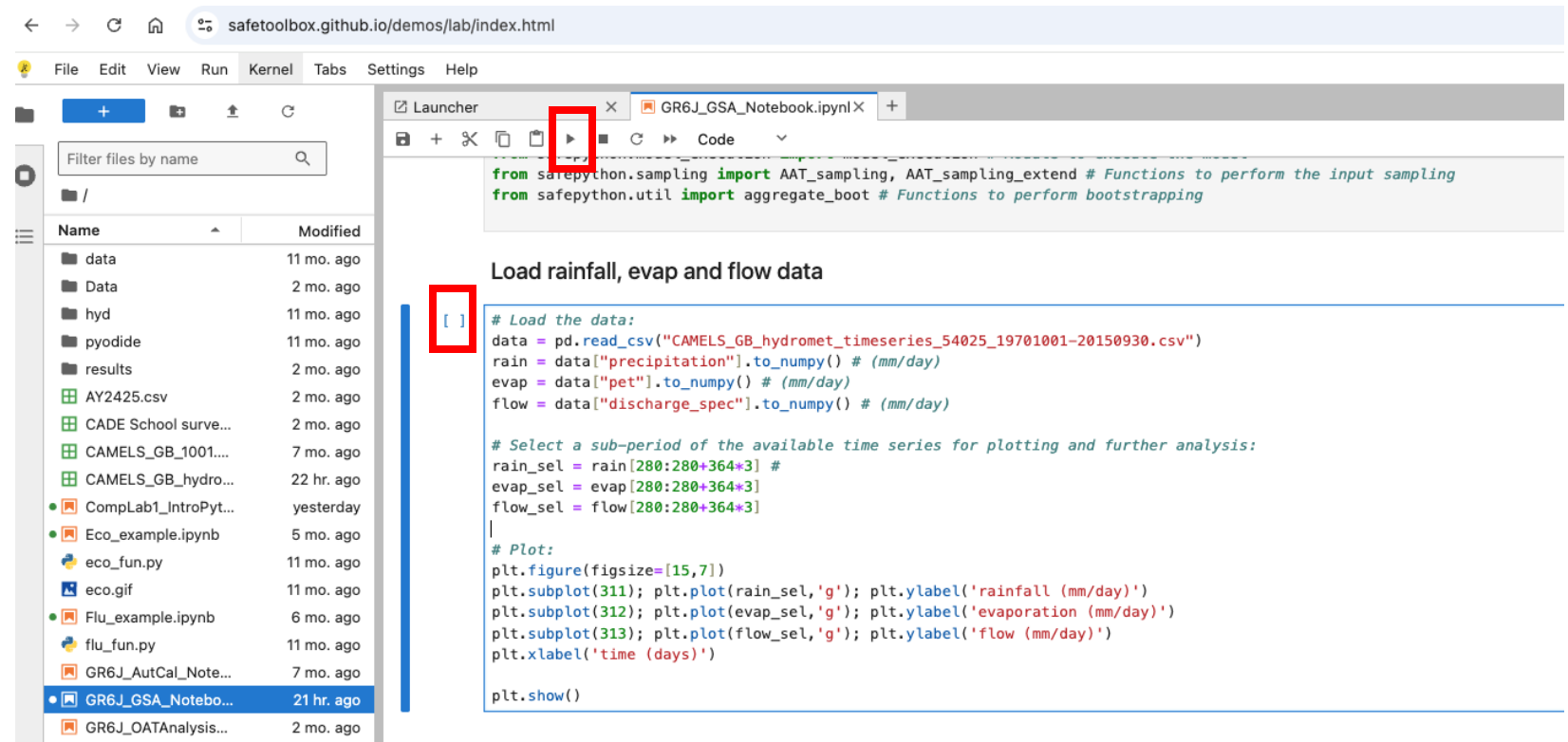
The 'GR6J_GSA_Notebook.ipynb', 'PerfMetrics.py', 'GR6J.py', and 'CAMELS_GB_hydro...01001-20150930.csv' files are selected. The 'Open' button is visible at the bottom right of the file explorer window.

3. Run the code

The Jupyter Notebook includes cells of executable code and cells of text with explanations of what is going on.

The cells of code can be executed one at the time by selecting the cell with the mouse and clicking the “run” icon at the top (or *shift+enter*).

While the code is running, an asterisk will appear in the brackets next to the cell code. If the code has been executed correctly, the asterisk will be replaced by a number. If the execution produces any plot or numerical results, they will appear under the cell



The screenshot shows a web-based Jupyter Notebook interface. The browser address bar displays `safetoolbox.github.io/demos/lab/index.html`. The interface includes a top menu bar with options: File, Edit, View, Run, Kernel, Tabs, Settings, and Help. On the left, a file browser pane shows a directory structure with files like `data`, `Data`, `hyd`, `pyodide`, `results`, and several CSV files. The main area displays a Jupyter Notebook with a tab titled `GR6J_GSA_Notebook.ipynb`. The notebook has a toolbar with icons for saving, opening, and running cells. The 'run' icon (a right-pointing triangle) is highlighted with a red box. Below the toolbar, a code cell is visible. The code cell's input prompt `[]` is highlighted with a red box. The code in the cell is as follows:

```
# Load the data:
data = pd.read_csv("CAMELS_GB_hydromet_timeseries_54025_19701001-20150930.csv")
rain = data["precipitation"].to_numpy() # (mm/day)
evap = data["pet"].to_numpy() # (mm/day)
flow = data["discharge_spec"].to_numpy() # (mm/day)

# Select a sub-period of the available time series for plotting and further analysis:
rain_sel = rain[280:280+364*3] #
evap_sel = evap[280:280+364*3]
flow_sel = flow[280:280+364*3]

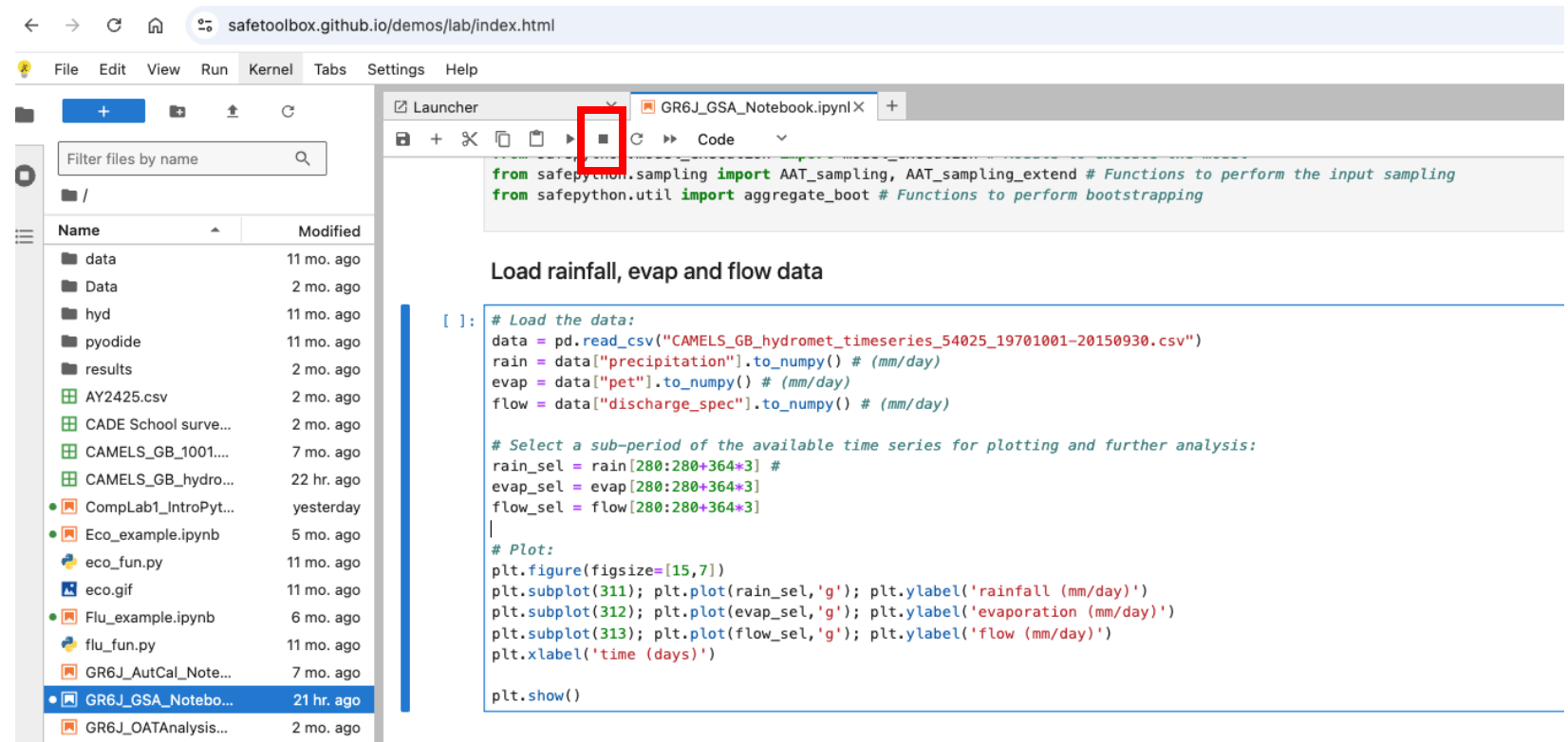
# Plot:
plt.figure(figsize=[15,7])
plt.subplot(311); plt.plot(rain_sel,'g'); plt.ylabel('rainfall (mm/day)')
plt.subplot(312); plt.plot(evap_sel,'g'); plt.ylabel('evaporation (mm/day)')
plt.subplot(313); plt.plot(flow_sel,'g'); plt.ylabel('flow (mm/day)')
plt.xlabel('time (days)')

plt.show()
```

3. Run the code - *Troubleshooting*

Normally all commands should run in less than a minute

If it takes longer than that, the system may be stuck and you may need to interrupt the execution and repeat



The screenshot shows the safetoolbox.github.io/demos/lab/index.html interface. On the left is a file explorer with a search bar and a list of files and folders. The right pane shows a code editor with a toolbar above it. A red box highlights the interrupt button (a square with a vertical line) in the toolbar. The code editor contains the following Python code:

```
from safepython.sampling import AAT_sampling, AAT_sampling_extend # Functions to perform the input sampling
from safepython.util import aggregate_boot # Functions to perform bootstrapping

Load rainfall, evap and flow data

[ ]: # Load the data:
data = pd.read_csv("CAMELS_GB_hydromet_timeseries_54025_19701001-20150930.csv")
rain = data["precipitation"].to_numpy() # (mm/day)
evap = data["pet"].to_numpy() # (mm/day)
flow = data["discharge_spec"].to_numpy() # (mm/day)

# Select a sub-period of the available time series for plotting and further analysis:
rain_sel = rain[280:280+364*3] #
evap_sel = evap[280:280+364*3]
flow_sel = flow[280:280+364*3]

# Plot:
plt.figure(figsize=[15,7])
plt.subplot(311); plt.plot(rain_sel,'g'); plt.ylabel('rainfall (mm/day)')
plt.subplot(312); plt.plot(evap_sel,'g'); plt.ylabel('evaporation (mm/day)')
plt.subplot(313); plt.plot(flow_sel,'g'); plt.ylabel('flow (mm/day)')
plt.xlabel('time (days)')

plt.show()
```