

Short Introduction to Jupyter and Python for Numerical Operations



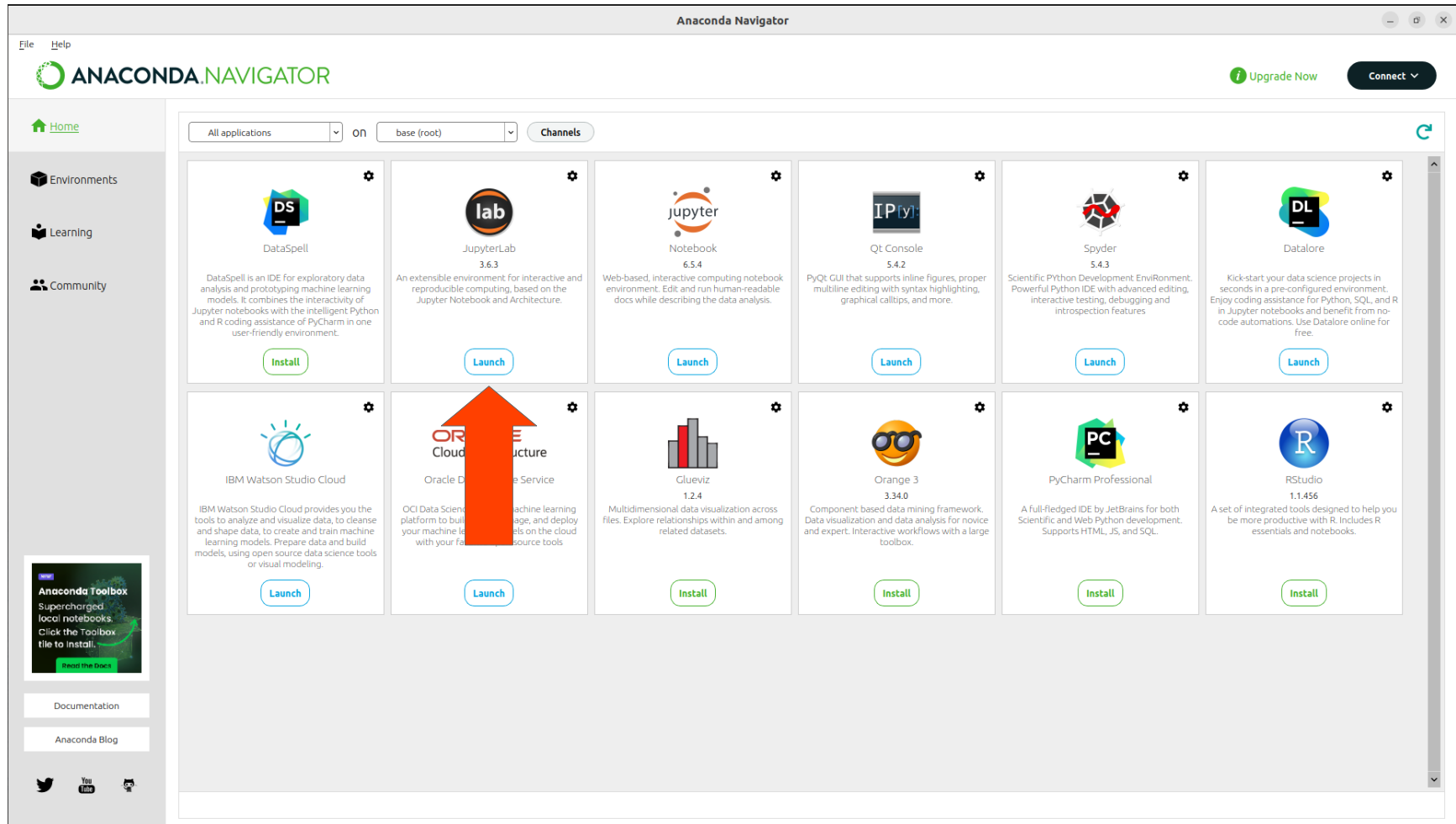
Why use Python? How does it work?

- Compared to Excel and Matlab, Python has several advantages
 - it is free and developed by a very active community (not a company)
 - it has a large number of specific modules (“packages”) that have been developed for specific applications (like numpy and scipy for numerical calculations)
 - it is easy to learn thanks to many available tutorials
- Python is a programming language that relies on “scripts”, that are text files with code written in successive lines
 - typical Python scripts end with suffix “.py”
- Anaconda is convenient to install Python on any computer (with Linux, Windows, Mac-OS) and handle packages (and dependencies when installing them)

Install Anaconda

- Go to <https://www.anaconda.com/download> or <https://docs.conda.io/projects/miniconda/en/latest/miniconda-install.html>
- Follow the instructions to install Anaconda or Miniconda on your computer
 - you may have to go to the folder where conda is installed to run in command line ``conda init`` for the system to know where conda is located
- Launch anaconda-navigator
- Launch jupyter-lab
 - notebooks are a mixture of code (like Python) and text / images that is very useful to didactic presentations
 - let's first see how to manipulate the notebook cells, then let's write Python code

Launch Jupyter Lab



Cells in Jupyter

The screenshot displays the Jupyter Notebook interface in a web browser. The address bar shows the URL: `localhost:8888/lab/tree/notebooks/basics_for_course/basic_python.ipynb`. The interface includes a top menu bar with options like File, Edit, View, Run, Kernel, Tabs, Settings, and Help. Below the menu is a toolbar with icons for file operations. On the left, a file browser sidebar shows the directory structure: `/notebooks/basics_for_course/`. It lists files with their names and last modified times: `basic_pyth...` (24 minutes ago), `basics_jup...` (5 months ago), `screenshot...` (33 minutes ago), and `screenshot...` (an hour ago). The main area displays a code cell titled "Short Introduction to Jupyter and Python for Numerical Operations". The cell contains a markdown header and a paragraph: "In Python, everything is an object, defined by a 'type'. For example, numbers can be integers or floats (among other types, but these are the most common types)." Below the text are five code input areas, each preceded by a prompt: `[1]: a = 1`, `[2]: type(a)`, `[2]: int`, `[3]: a = 1.0`, and `[4]: type(a)`. The last line of code is `[4]: float`. The bottom status bar shows the current mode as "Simple", the kernel as "Python 3 (ipykernel)", and the file name as "basic_python.ipynb".

localhost:8888/lab/tree/notebooks/basics_for_course/basic_python.ipynb

File Edit View Run Kernel Tabs Settings Help

Filter files by name

/notebooks/basics_for_course/

Name	Last Modified
basic_pyth...	24 minutes ago
basics_jup...	5 months ago
screenshot...	33 minutes ago
screenshot...	an hour ago

Short Introduction to Jupyter and Python for Numerical Operations

In Python, everything is an object, defined by a 'type'. For example, numbers can be integers or floats (among other types, but these are the most common types).

```
[1]: a = 1
```

```
[2]: type(a)
```

```
[2]: int
```

```
[3]: a = 1.0
```

```
[4]: type(a)
```

```
[4]: float
```

Simple 0 3 Python 3 (ipykernel) | Idle Mode: Command Ln 3, Col 66 basic_python.ipynb

Cells in Jupyter

The screenshot shows the JupyterLab interface in a web browser. The address bar indicates the URL is `localhost:8888/lab/tree/notebooks/basics_for_course/basic_python.ipynb`. The left sidebar contains a file browser with a search bar and a list of files in the `/notebooks/basics_for_course/` directory. The file `basic_pyth...` is highlighted. A pink callout bubble points to this file with the text "list of files (double click to open)". The main area displays the content of the selected notebook, titled "Short Introduction to Jupyter and Python for Numerical Operations". The notebook content includes a text cell explaining that in Python, everything is an object defined by a 'type', and a series of code cells demonstrating variable assignment and type checking.

File browser contents:

Name	Last Modified
basic_pyth...	24 minutes ago
basics_jup...	5 months ago
screenshot...	33 minutes ago
screenshot...	an hour ago

Notebook content:

Short Introduction to Jupyter and Python for Numerical Operations

In Python, everything is an object, defined by a 'type'. For example, numbers can be integers or floats (among other types, but these are the most common types).

```
[1]: a = 1
```

```
[2]: type(a)
```

```
[2]: int
```

```
[3]: a = 1.0
```

```
[4]: type(a)
```

```
[4]: float
```

Bottom status bar: Simple 0 3 Python 3 (ipykernel) | Idle Mode: Command Ln 3, Col 66 basic_python.ipynb

Cells in Jupyter

The screenshot displays the JupyterLab web interface in a browser window at `localhost:8888/lab/tree/notebooks/basics_for_course/basic_python.ipynb`. The interface includes a top menu bar (File, Edit, View, Run, Kernel, Tabs, Settings, Help) and a toolbar with various icons. On the left, a file browser sidebar shows a directory structure with files like `basic_py`, `basics_ju`, `screensho`, and `screenshot`. The main area shows an open notebook with the title "Short Introduction to Jupyter and Python for Numerical Operations". The notebook content includes a text cell explaining that in Python, everything is an object defined by a 'type', followed by four code cells demonstrating variable assignment and type checking:

```
[1]: a = 1
[2]: type(a)
[2]: int
[3]: a = 1.0
[4]: type(a)
[4]: float
```

A red callout bubble points to the '+' button in the top-left toolbar, with the text "create a new notebook".

At the bottom of the interface, the status bar shows "Simple", "0", "3", "Python 3 (ipykernel) | Idle", "Mode: Command", "Ln 3, Col 66", and "basic_python.ipynb".

Cells in Jupyter

The screenshot displays the JupyterLab web interface in a browser at `localhost:8888/lab/tree/notebooks/basics_for_course/basic_python.ipynb`. The interface is divided into three main sections:

- File Browser (Left):** Shows a directory structure under `/notebooks/basics_for_course/`. A table lists files with their names and last modified times:

Name	Last Modified
• basic_pyth...	24 minutes ago
basics_jup...	5 months ago
screenshot...	33 minutes ago
screenshot...	an hour ago

- Tab Bar (Top):** Shows three open notebooks: `nb_neuronal_dynamics2.ipynb`, `nb_neuronal_dynamics1.ipynb`, and `basic_python.ipynb`. The `basic_python.ipynb` tab is active and highlighted.
- Notebook Editor (Right):** Displays the content of `basic_python.ipynb`. It starts with a title cell:

Short Introduction to Jupyter and Python for

. Below it is a text cell:

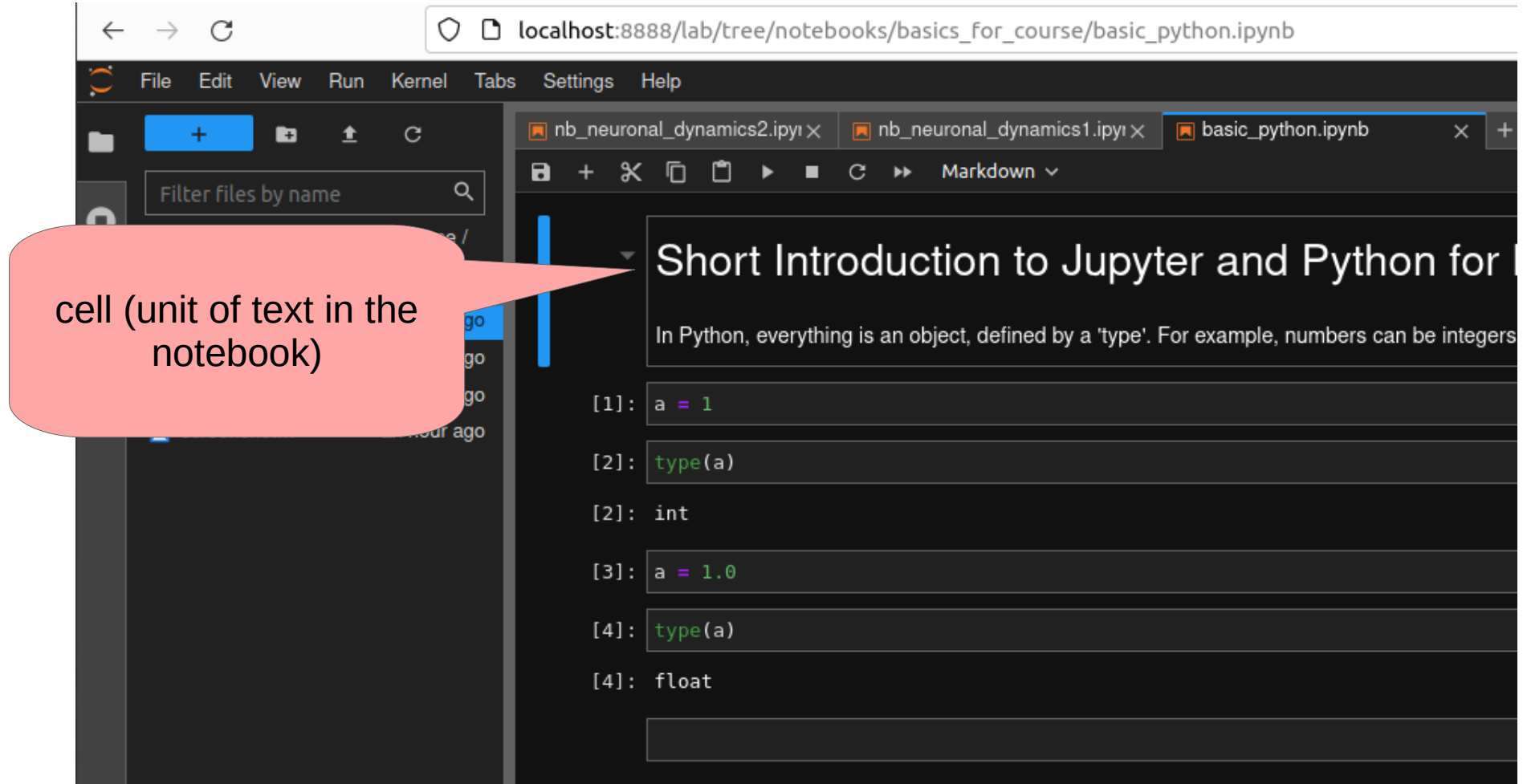
In Python, everything is an object, defined by a 'type'. For example, numbers can be integers

. This is followed by four code cells:

```
[1]: a = 1
[2]: type(a)
[2]: int
[3]: a = 1.0
[4]: type(a)
[4]: float
```

A red callout bubble points to the `basic_python.ipynb` tab with the text: **current file being edited (here a notebook)**.

Cells in Jupyter



The image shows a Jupyter Notebook interface in a web browser. The address bar displays `localhost:8888/lab/tree/notebooks/basics_for_course/basic_python.ipynb`. The top menu bar includes `File`, `Edit`, `View`, `Run`, `Kernel`, `Tabs`, `Settings`, and `Help`. Below the menu, there are icons for file operations and a search bar labeled "Filter files by name". The notebook has three tabs open: `nb_neuronal_dynamics2.ipynb`, `nb_neuronal_dynamics1.ipynb`, and `basic_python.ipynb`. The active tab, `basic_python.ipynb`, shows a cell with the title "Short Introduction to Jupyter and Python for I" and the text "In Python, everything is an object, defined by a 'type'. For example, numbers can be integers". Below the text are several code cells with the following content:

```
[1]: a = 1
```

```
[2]: type(a)
```

```
[2]: int
```

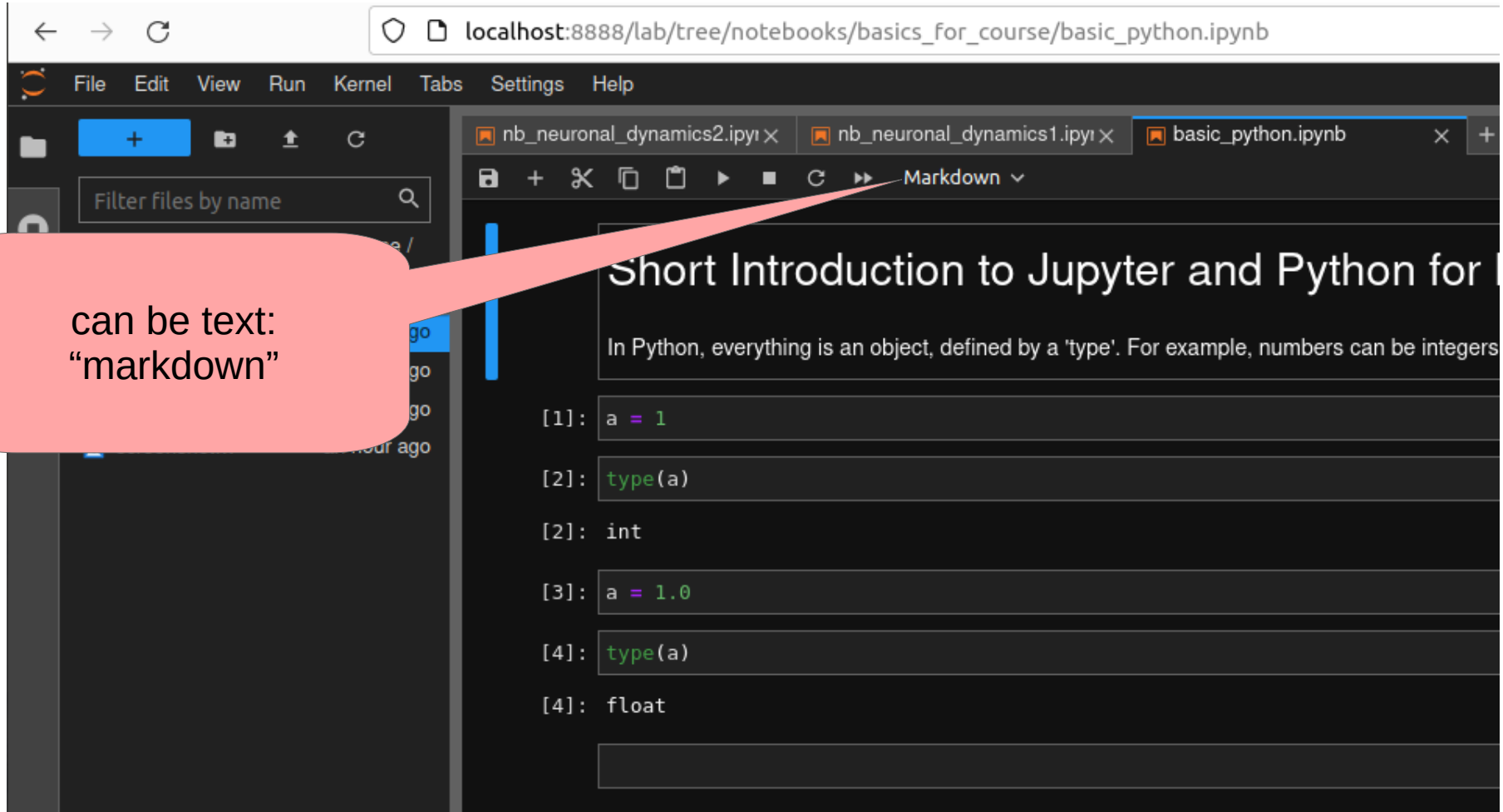
```
[3]: a = 1.0
```

```
[4]: type(a)
```

```
[4]: float
```

A red callout bubble points to the first cell (the text cell) and contains the text: "cell (unit of text in the notebook)".

Cells in Jupyter



The screenshot shows a Jupyter Notebook interface in a web browser. The address bar displays `localhost:8888/lab/tree/notebooks/basics_for_course/basic_python.ipynb`. The notebook has three tabs: `nb_neuronal_dynamics2.ipynb`, `nb_neuronal_dynamics1.ipynb`, and `basic_python.ipynb`. The `basic_python.ipynb` tab is active, showing a cell type dropdown menu with `Markdown` selected. The cell content includes a heading `Short Introduction to Jupyter and Python for` and a paragraph `In Python, everything is an object, defined by a 'type'. For example, numbers can be integers`. Below the text are four code cells:

```
[1]: a = 1
```

```
[2]: type(a)
```

```
[2]: int
```

```
[3]: a = 1.0
```

```
[4]: type(a)
```

```
[4]: float
```

A red callout bubble points to the `Markdown` option in the cell type dropdown menu, containing the text:

can be text:
"markdown"

Cells in Jupyter

The screenshot shows the JupyterLab web interface. The browser address bar displays `localhost:8888/lab/tree/notebooks/basics_for_course/basic_python.ipynb`. The top menu bar includes `File`, `Edit`, `View`, `Run`, `Kernel`, `Tabs`, `Settings`, and `Help`. The left sidebar shows a file browser with a search bar labeled "Filter files by name" and a directory path `/ notebooks / basics_for_course /`. The main area displays a notebook with three tabs: `nb_neuronal_dynamics2.ipynb`, `nb_neuronal_dynamics1.ipynb`, and `basic_python.ipynb`. The `basic_python.ipynb` tab is active, showing a code cell with the following content:

```
# Short Introduction to Jupyter and Python for Numerical Operations)

In Python, everything is an object, defined by a 'type'. For example, num

[1]: a = 1

[2]: type(a)

[2]: int

[3]: a = 1.0

[4]: type(a)

[4]: float
```

A red callout bubble with the text "double click to edit" points to the first code cell. The notebook interface also shows a toolbar with icons for saving, adding, deleting, and running cells, as well as a dropdown menu for the cell type (currently set to "Markdown").

Cells in Jupyter

The screenshot shows the JupyterLab web interface in a browser at `localhost:8888/lab/tree/notebooks/basics_for_course/basic_python.ipynb`. The interface includes a top menu bar (File, Edit, View, Run, Kernel, Tabs, Settings, Help) and a left sidebar with a file browser. The main area displays a notebook with a title "Short Introduction to Jupyter and Python for I" and a text cell containing the sentence "In Python, everything is an object, defined by a 'type'. For example, numbers can be integers". Below the text are four code cells:

```
[1]: a = 1
```

```
[2]: type(a)
```

```
[2]: int
```

```
[3]: a = 1.0
```

```
[4]: type(a)
```

```
[4]: float
```

A red callout bubble points to the run button (a right-pointing triangle) in the toolbar, with the text "then run (or MAJ+ENTER)".

Cells in Jupyter

can be “code”, running
the cell performs the
calculation

The screenshot displays the Jupyter Notebook web interface in a browser. The address bar shows the URL: `localhost:8888/lab/tree/notebooks/basics_for_course/basic_python.ipynb`. The top menu bar includes `File`, `Edit`, `View`, `Run`, `Kernel`, `Tabs`, `Settings`, and `Help`. Below the menu is a toolbar with icons for file operations and execution. The left sidebar shows a file explorer with a search bar labeled "Filter files by name". The main area shows a notebook with three tabs: `nb_neuronal_dynamics2.ipynb`, `nb_neuronal_dynamics1.ipynb`, and `basic_python.ipynb`. The `basic_python.ipynb` tab is active, showing a code cell with the following content:

```
[1]: a = 1
[2]: type(a)
[2]: int
[3]: a = 1.0
[4]: type(a)
[4]: float
```

A red callout bubble points to the `Code` button in the toolbar, indicating that the cell can be set to "code" mode to perform calculations.

Cells in Jupyter

The screenshot shows the JupyterLab interface. The browser address bar displays `localhost:8888/lab/tree/notebooks/basics_for_course/basic_python.ipynb`. The top menu bar includes `File`, `Edit`, `View`, `Run`, `Kernel`, `Tabs`, `Settings`, and `Help`. On the left, a file browser sidebar shows the directory `/ notebooks / basics_for_course /` with a table of files:

Name	Last Modified
basic_pyth...	a minute ago

The main area displays the `basic_python.ipynb` notebook. The toolbar at the top of the notebook area includes icons for saving, creating a new cell, deleting a cell, copying, pasting, running, and other actions. A red callout bubble points to the '+' icon in the toolbar, with the text "create a new cell". The notebook content shows a code cell with the following text:

```
# Short Introduction to Jupyter and Python for Numerical Operations)

In Python, everything is an object, defined by a 'type'. For example, num

[1]: 
[2]: t
[2]: int
[3]: 
[4]: 
[4]: float
```

Cells in Jupyter

The screenshot displays the JupyterLab web interface. The browser address bar shows the URL: `localhost:8888/lab/tree/notebooks/basics_for_course/basic_python.ipynb`. The interface includes a top menu bar with options: File, Edit, View, Run, Kernel, Tabs, Settings, and Help. On the left, a file browser sidebar shows the directory structure: `/ notebooks / basics_for_course /`. A table lists files with columns for Name and Last Modified. The file `basic_pyth...` is highlighted, showing it was modified 'a minute ago'. The main area on the right is the code editor for `basic_python.ipynb`. It features a toolbar with icons for saving, creating new cells, deleting cells, copying, pasting, running, and interrupting the kernel. A pink callout bubble with the text 'cut-copy-paste' points to the copy icon in this toolbar. The code editor shows a markdown cell with the text 'Start Introduction to Jupyter and Python for Numerical Operations)' and a code cell with the following Python code:

```
[1]: a =  
[2]: type(  
[2]: int  
[3]:  
[4]:  
[4]: float
```


Useful Tips

- Shortcut MAJ+ENTER to run current cell
- Menu 'Cells': 'Run All Cells' to run the whole notebook
 - the memory is updated by the execution of each cell, so if you don't run cells in order (or just go to a cell after opening the notebook), the variables in memory may not be what you think...
- Restart the kernel when something went wrong in the notebook (like no cell output although there should be...)

Create Environment with Selected Packages

- Back to anaconda-navigator: follow the following steps described in the following slides
- Go to 'Environments' tab and create a new environment
 - name it for example 'env1' and go with the latest version of Python as proposed
- Search packages that we are going to use and install them:
 - 'numpy' and 'scipy' for calculation
 - 'matplotlib' for visualization
 - 'jupyter lab'

Create Environment and Select Packages Manually

File

Help

ANACONDA.NAVIGATOR

Upgrade Now

Connect

Home

Environments

Learning Center

Community

base (root)

brian2

course

jax

mne

nilearn

pymc5

pytorch

sktime

Anaconda Toolbox

Documentation

Anaconda Blog

Twitter

YouTube

GitHub

Search Environments

base (root)

brian2

course

jax

mne

nilearn

pymc5

pytorch

sktime

Installed

Channels

Update index...

Search Packages

Name	T	Description	Version
<input checked="" type="checkbox"/> _libgcc_mutex	<input type="radio"/>	Mutex for libgcc and libgcc-ng	0.1
<input checked="" type="checkbox"/> _openmp_mutex	<input type="radio"/>	Openmp implementation mutex	5.1
<input checked="" type="checkbox"/> aiofiles	<input type="radio"/>	File support for asyncio	22.1.0
<input checked="" type="checkbox"/> aiosqlite	<input type="radio"/>		0.18.0
<input checked="" type="checkbox"/> alabaster	<input type="radio"/>	Configurable, python 2+3 compatible sphinx theme.	0.7.12
<input checked="" type="checkbox"/> anaconda-client	<input type="radio"/>	Anaconda.org command line client library	1.11.2
<input checked="" type="checkbox"/> anaconda-project	<input type="radio"/>	Tool for encapsulating, running, and reproducing data science projects	0.11.1
<input checked="" type="checkbox"/> anyio	<input type="radio"/>	High level compatibility layer for multiple asynchronous event loop implementations on python	3.5.0
<input checked="" type="checkbox"/> archspec	<input type="radio"/>	A library to query system architecture	0.2.1
<input checked="" type="checkbox"/> argon2-cffi	<input type="radio"/>	The secure argon2 password hashing algorithm.	21.3.0
<input checked="" type="checkbox"/> argon2-cffi-bindings	<input type="radio"/>	Low-level python cffi bindings for argon2	21.2.0
<input checked="" type="checkbox"/> arrow	<input type="radio"/>	Better dates & times for python	1.2.3
<input checked="" type="checkbox"/> astroid	<input type="radio"/>	A abstract syntax tree for python with inference support.	2.14.2
<input checked="" type="checkbox"/> asttokens	<input type="radio"/>	The asttokens module annotates python abstract syntax trees (asts) with the positions of tokens and text in the source code that generated them.	2.0.5
<input checked="" type="checkbox"/> atomicwrites	<input type="radio"/>	Atomic file writes	1.4.0
<input checked="" type="checkbox"/> attrs	<input type="radio"/>	Attrs is the python package that will bring back the joy of writing classes by relieving you from the drudgery of implementing object protocols (aka dunder methods).	23.1.0
<input checked="" type="checkbox"/> autopep8	<input type="radio"/>	A tool that automatically formats python code to conform to the pep 8 style guide	1.6.0
<input checked="" type="checkbox"/> babel	<input type="radio"/>	Utilities to internationalize and localize python applications	2.11.0
<input checked="" type="checkbox"/> backcall	<input type="radio"/>	Specifications for callback functions passed in to an api	0.2.0

362 packages available

Create

Clone

Import

Backup

Remove

Create Environment and Select Packages Manually

Anaconda Navigator

File Help

ANACONDA.NAVIGATOR

Upgrade Now Connect

Home

Environments

Learning

Community

base (root)

brian2

course

jax

mne

nilearn

pymc5

pytorch

sktime

Search Environments

Installed Channels Update index...

Search Packages

Name	Description	Version
✓ _libgcc_mutex	○ Mutex for libgcc and libgcc-ng	0.1
✓ _openmp_mutex	○ Openmp implementation mutex	5.1
✓ aiofiles	○ File support for asyncio	22.1.0
✓ aiosqlite	○	0.18.0
✓ alabaster	○ Configurable, python 2+3 compatible sphinx theme.	0.7.12
✓ anaconda-client	○ Anaconda.org command line client library	1.11.2
✓ anaconda-project	○ Tool for encapsulating, running, and reproducing data science projects	0.11.1
✓ anyio	○ High level compatibility layer for multiple asynchronous event loop implementations on python	3.5.0
✓ archspec	○ A library to query system architecture	0.2.1
✓ argon2-cffi	○ The secure argon2 password hashing algorithm.	21.3.0
✓ argon2-cffi-bindings	○ Low-level python cffi bindings for argon2	21.2.0
✓ arrow	○ Better dates & times for python	1.2.3
✓ astroid	○ A abstract syntax tree for python with inference support.	2.14.2
✓ asttokens	○ The asttokens module annotates python abstract syntax trees (asts) with the positions of tokens and text in the source code that generated them.	2.0.5
✓ atomicwrites	○ Atomic file writes	1.4.0
✓ attrs	○ Attrs is the python package that will bring back the joy of writing classes by relieving you from the drudgery of implementing object protocols (aka dunder methods).	23.1.0
✓ autopep8	○ A tool that automatically formats python code to conform to the pep 8 style guide	1.6.0
✓ babel	○ Utilities to internationalize and localize python applications	2.11.0
✓ backcall	○ Specifications for callback functions passed in to an api	0.2.0


362 packages available

Anaconda Toolbox
Supercharged local notebooks.
Click the Toolbox tile to install.
Install the Toolbox

Documentation

Anaconda Blog

Create Clone Import Backup Remove



Create Environment and Select Packages Manually

Anaconda Navigator

File Help

ANACONDA.NAVIGATOR

Upgrade Now Connect

Home

Environments

Learning

Community

Search Environments

base (root)

brian2

course

emn

jax

mine

nilearn

pymc5

pytorch

sktime

All Channels Update index...

matplotlib

Name	Description	Version
<input type="checkbox"/> basemap	Plot on map projections using matplotlib	1.3.6
<input type="checkbox"/> basemap-data	Plot on map projections (with coastlines and political boundaries) using matplotlib.	1.3.6
<input type="checkbox"/> basemap-data-hires	Plot on map projections (with coastlines and political boundaries) using matplotlib.	1.3.6
<input type="checkbox"/> descartes	Use geometric objects as matplotlib paths and patches.	1.1.0
<input type="checkbox"/> ipympl	Matplotlib jupyter	0.9.3
<input checked="" type="checkbox"/> matplotlib		3.8.0
<input type="checkbox"/> matplotlib-base	Publication quality	3.8.0
<input type="checkbox"/> matplotlib-inline	Inline matplotlib	0.1.6
<input type="checkbox"/> mpl-scatter-density	Matplotlib helpers to make density scatter plots	0.7
<input type="checkbox"/> mpld3	D3 viewer for matplotlib.	0.5.9

1) search

2) check box

3) press 'apply' to install

10 packages available matching "matplotlib" 1 package selected

Apply Clear

Anaconda Toolbox
Supercharged local notebooks.
Click the Toolbox tile to install.
Install the Toolbox

Documentation

Anaconda Blog

Create Clone Import Backup Remove

Create Environment Using YAML File

Anaconda Navigator

File Help

ANACONDA.NAVIGATOR

Upgrade Now Connect

Home

Environments

Learning

Community

Search Environments

base (root)

brian2

course

jax

mne

nilearn

pymc5

pytorch

sktime

Installed Channels Update index...

Search Packages

Name	Description	Version
✓ _libgcc_mutex	○ Mutex for libgcc and libgcc-ng	0.1
✓ _openmp_mutex	○ Openmp implementation mutex	5.1
✓ aiofiles	○ File support for asyncio	22.1.0
✓ aiosqlite	○	0.18.0
✓ alabaster	○ Configurable, python 2+3 compatible sphinx theme.	0.7.12
✓ anaconda-client	○ Anaconda.org command line client library	1.11.2
✓ anaconda-project	○ Tool for encapsulating, running, and reproducing data science projects	0.11.1
✓ anyio	○ High level compatibility layer for multiple asynchronous event loop implementations on python	3.5.0
✓ archspec	○ A library to query system architecture	0.2.1
✓ argon2-cffi	○ The secure argon2 password hashing algorithm.	21.3.0
✓ argon2-cffi-bindings		21.2.0
✓ arrow		1.2.3
✓ astroid		2.14.2
✓ asttokens		2.0.5
✓ atomicwrites	○ Atomic file writes	1.4.0
✓ attrs	○ Attrs is the python package that will bring back the joy of writing classes by relieving you from the drudgery of implementing object protocols (aka dunder methods).	23.1.0
✓ autopep8	○ A tool that automatically formats python code to conform to the pep 8 style guide	1.6.0
✓ babel	○ Utilities to internationalize and localize python applications	2.11.0
✓ backcall	○ Specifications for callback functions passed in to an api	0.2.0

362 packages available

Import from environment.yaml

Create Clone Import Backup Remove

Anaconda Toolbox
Supercharged local notebooks.
Click the Toolbox tile to install.
Install the Toolbox

Documentation

Anaconda Blog

Twitter YouTube GitHub

Now to Practice with Python Programming

- Download and open the notebook 'basic_python.ipynb' in folder 'basics/python/'
- For further information about notebooks, check <https://docs.jupyter.org/en/latest/>
 - markdown syntax: <https://fr.wikipedia.org/wiki/Markdown>
- Numpy documentation: <https://numpy.org/doc/stable/reference/>
 - tutorial: https://numpy.org/devdocs/user/absolute_beginners.html
- Scipy documentation: <https://docs.scipy.org/doc/scipy/reference/index.html>
- Matplotlib tutorial: https://matplotlib.org/stable/users/explain/quick_start.html