The simulation is a typical biotechnological project whose experiments were imitated. The aim is to optimise the expression of a protein in a bacterium. To do this, simulated experiments must be carried out. First, the production strain will be characterised by temperature and biomass. Then the correct promoter sequence must be designed and cloned. ~~In the end~~, the production must be above a certain limit in order to be economically viable. This simulates some important data and decisions in biotechnology.

The program is based on a Python Jupyter Notebook in which Python program code and text/images are combined. The main page to work on, which should also be loaded automatically, is 1-Laboratory.ipynb. You just have to work through it from top to bottom. ~~There the code cells are adjusted~~ in such a way that the simulations are successful.
The Notebooks are composed of a sequence of cells that can be either text or python code cells to be run. Code cells have a grey background and after execution the output is shown directly beneath. A blue stripe on the left edge of the screen marks the currently active cell. To activate a cell below or above, use the arrow down and up keys in Command mode.
Code cells can be edited and executed multiple times. Next to the code cells in the upper left corner the status of the cell is displayed in square brackets. If you have not executed a cell yet, the brackets are empty. If the computer is currently executing the code, an small star appears there. If the cell has been executed, a number corresponding to the execution order of the cells is shown.
More detailed information on how to use Jupyter Notebooks are given below in the Jupyter Notebook Cheat Sheet, which you will also find in the “Python and Jupyter Notebook basics” folder.

• edureka! •

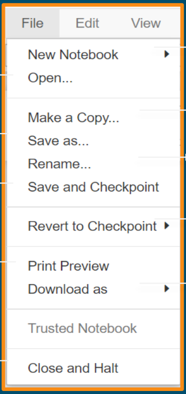
JUPYTER NOTEBOOK CHEAT SHEET

Learn PYTHON from experts at https://www.edureka.co

Jupyter Notebook

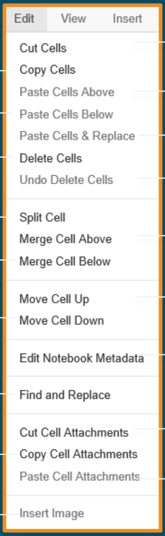
Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. It is used for data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

Saving/Loading Notebook



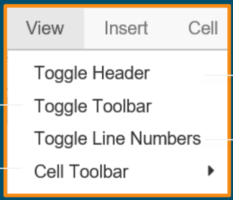
- Create new Notebook
- Open an existing Notebook
- Make copy of the current Notebook
- Save Current Notebook
- Rename current Notebook
- Save Current Notebook & record Checkpoint
- Revert Notebook to a previous Checkpoint
- Preview of the printed Notebook
- Download Notebook as-IPython Notebook
  - Python
  - HTML
  - Markdown
  - PDF
- Close Notebook & stop running scripts

Edit Cells



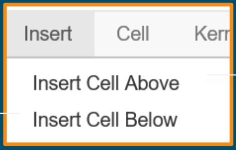
- Cut the selected cells to Clipboard
- Copy cells from Clipboard to current position
- Paste cells above current cell
- Paste cells below current cell
- Paste cells on top of current cell
- Delete cells
- Revert 'Delete cells' invocation
- Split up cell from current position
- Merge current cell with above
- Merge current cell with below
- Move current cell up
- Move current cell down
- Adjust Metadata underlying the current Notebook
- Find and replace in selected cells
- Remove cell attachments
- Copy attachments of current cell
- Paste attachments of current cell
- Insert image in selected cells

View Cells



- Toggle display of Jupyter logo & Filename
- Toggle display of Toolbar
- Toggle line numbers in cell
- Toggle display of cell action icons

Insert Cells



- Add new cell above the current one
- Add new cell below the current one

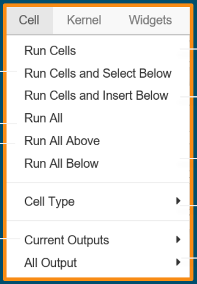
Keyboard Shortcuts

Command	Description
enter	enter edit mode
Command + a; Command + c; Command + v	select all; copy; paste
Command + z; Command + y	undo; redo
Command + s	save and checkpoint
Command + b; Command + a	insert cell below; insert cell above
Shift + Enter	run cell, select below
Shift + m	merge cells
Command +]; Command + [	indent; dedent
Ctrl + Enter	run cell
Option + Return	run cell, insert cell below
Escape	enter command mode
Escape + d + d	delete selected cell
Escape + y	change cell to code
Escape + m	change cell to markdown
Escape + r	change cell to raw
Escape + 1	change cell to Heading 1
Escape + n	change cell to heading n
Escape + b	create cell below
Escape + a	Insert cell above

Magic Commands

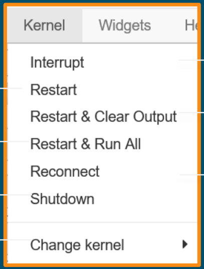
Statement	Explanation	Example
%magic	Comprehensively lists and explains magic functions	%magic
%automagic	When active, enables you to call magic functions without the '%'	%automagic
%quickref	Launch IPython quick reference	%quickref
%pastebin	Pastebins lines from your current session.	%pastebin 3 18-20 ~1/1-5
%debug	Enters the interactive debugger	%debug
%hist	Print command input and output history	%hist
%pdb	Automatically enter python debugger after any exception	%pdb
%cpaste	Opens up a special prompt for manually pasting Python code for execution	%cpaste
%reset	Delete all variables and names defined in the current namespace	%reset
%run	Run a python script inside a notebook	%run script.py
%who, %who_ls, %whos	Display variables defined in the interactive namespace, with varying levels of verbosity	%who, %who_ls, %whos
%xdel	Delete a variable in the local namespace. Clear any references to that variable	%xdel variable
%time	Times a single statement	In [561]: %time method = [a for a in data if b.startswith('http')]

Execute Cells



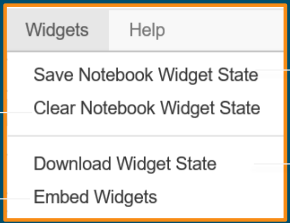
- Run Selected Cells
- Run Current Cells down & create one below
- Run Current Cells down & create one above
- Run all Cells
- Run all Cells above the current one
- Run all Cells below current one
- Change the cell type
- Toggle & clear current outputs
- Toggle & clear all outputs

Kernel Cells



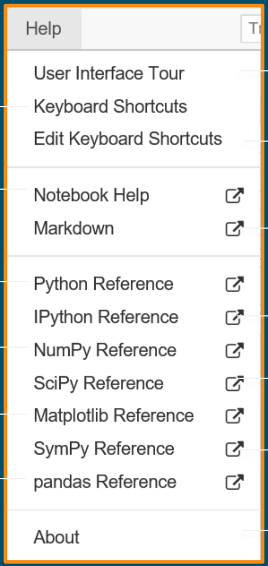
- Interrupt kernel
- Restart Kernel
- Interrupt kernel & Clear all output
- Restart Kernel & Run all cells
- Reconnect to a remote Notebook
- Shutdown all cells
- Run other installed kernels

Widgets



- Save Notebook with Interactive widget
- Clear Notebook with Interactive widget
- Download all widget models in use
- Embed current widgets

Help



- Walk through a UI Tour
- Built-in keyboard shortcuts
- Edit the Built-in keyboard shortcuts
- Notebook help topics
- Markdown available in Notebook
- Python help topics
- IPython help topics
- NumPy help topics
- SciPy help topics
- Matplotlib help topics
- SymPy help topics
- Pandas help topics
- About Jupyter Notebook

To understand how the code cells need to be adapted, you first have to understand the basics of Python programming code. To do this, take a look at the following Python Basics Cheat Sheet, which you will also find in the “Python and Jupyter Notebook basics” folder.
Additionally you should know that comments to the code lines are introduced with a # and go to the end of the line.
In the code cells you will also find functions that need to be executed. It is important to know that the parameters required for the function are passed to the function in the form of a comma-separated list in round brackets directly after the function name. These parameters can be defined using variables before executing the function.

Python For Data Science *Cheat Sheet*

Python Basics

Learn More Python for Data Science Interactively at www.datacamp.com

Variables and Data Types

Variable Assignment

```
>>> x=5
>>> x
 5
```

Calculations With Variables

`>>> x+2`
`7`
	Sum of two variables

`>>> x-2`
`3`
	Subtraction of two variables

`>>> x*2`
`10`
	Multiplication of two variables

`>>> x**2`
`25`
	Exponentiation of a variable

`>>> x%2`
`1`
	Remainder of a variable

`>>> x/float(2)`
`2.5`
	Division of a variable

Types and Type Conversion

`str()`	`'5', '3.45', 'True'`	Variables to strings
`int()`	`5, 3, 1`	Variables to integers
`float()`	`5.0, 1.0`	Variables to floats
`bool()`	`True, True, True`	Variables to booleans

Asking For Help

```
>>> help(str)
```

Lists

Also see NumPy Arrays

```
>>> a = 'is'
>>> b = 'nice'
>>> my_list = ['my', 'list', a, b]
>>> my_list2 = [[4,5,6,7], [3,4,5,6]]
```

Selecting List Elements

Index starts at 0

Subset

`>>> my_list[1]`	Select item at index 1
`>>> my_list[-3]`	Select 3rd last item

Slice

`>>> my_list[1:3]`	Select items at index 1 and 2
`>>> my_list[1:]`	Select items after index 0
`>>> my_list[:3]`	Select items before index 3
`>>> my_list[:]`	Copy my_list

Subset Lists of Lists

`>>> my_list2[1][0]`	my_list[list][itemOfList]
`>>> my_list2[1][:2]`	

List Operations

```
>>> my_list + my_list
['my', 'list', 'is', 'nice', 'my', 'list', 'is', 'nice']
>>> my_list * 2
['my', 'list', 'is', 'nice', 'my', 'list', 'is', 'nice']
>>> my_list2 > 4
True
```

List Methods

`>>> my_list.index(a)`	Get the index of an item
`>>> my_list.count(a)`	Count an item
`>>> my_list.append('!')`	Append an item at a time
`>>> my_list.remove('!')`	Remove an item
`>>> del(my_list[0:1])`	Remove an item
`>>> my_list.reverse()`	Reverse the list
`>>> my_list.extend('!')`	Append an item
`>>> my_list.pop(-1)`	Remove an item
`>>> my_list.insert(0,'!')`	Insert an item
`>>> my_list.sort()`	Sort the list

Strings

```
>>> my_string = 'thisStringIsAwesome'
>>> my_string
'thisStringIsAwesome'
```

String Operations

```
>>> my_string * 2
 'thisStringIsAwesomethisStringIsAwesome'
>>> my_string + 'Innit'
 'thisStringIsAwesomeInnit'
>>> 'm' in my_string
 True
```

String Operations

Index starts at 0

```
>>> my_string[3]
>>> my_string[4:9]
```

String Methods

`>>> my_string.upper()`	String to uppercase
`>>> my_string.lower()`	String to lowercase
`>>> my_string.count('w')`	Count String elements
`>>> my_string.replace('e', 'i')`	Replace String elements
`>>> my_string.strip()`	Strip whitespaces

Libraries

Import libraries

```
>>> import numpy
>>> import numpy as np
```

Selective import

```
>>> from math import pi
```

pandas	Data analysis
learn	Machine learning
NumPy	Scientific computing
matplotlib	2D plotting

Install Python

ANACONDA	spyder	jupyter
Leading open data science platform powered by Python	Free IDE that is included with Anaconda	Create and share documents with live code, visualizations, text, ...

Numpy Arrays

Also see Lists

```
>>> my_list = [1, 2, 3, 4]
>>> my_array = np.array(my_list)
>>> my_2darray = np.array([[1,2,3],[4,5,6]])
```

Selecting Numpy Array Elements

Index starts at 0

Subset

`>>> my_array[1]`
`2`
	Select item at index 1

Slice

`>>> my_array[0:2]`
`array([1, 2])`
	Select items at index 0 and 1

Subset 2D Numpy arrays

`>>> my_2darray[:,0]`
`array([1, 4])`
	my_2darray[rows, columns]

Numpy Array Operations

```
>>> my_array > 3
  array([False, False, False,  True], dtype=bool)
>>> my_array * 2
  array([2, 4, 6, 8])
>>> my_array + np.array([5, 6, 7, 8])
  array([6, 8, 10, 12])
```

Numpy Array Functions

`>>> my_array.shape`	Get the dimensions of the array
`>>> np.append(other_array)`	Append items to an array
`>>> np.insert(my_array, 1, 5)`	Insert items in an array
`>>> np.delete(my_array,[1])`	Delete items in an array
`>>> np.mean(my_array)`	Mean of the array
`>>> np.median(my_array)`	Median of the array
`>>> my_array.corrcoef()`	Correlation coefficient
`>>> np.std(my_array)`	Standard deviation

During the simulation you have to determine some parameters in different experiments, so that in the end the simulation is successful.

This includes the optimal growth temperature of your strain. It will be randomly initiated by the system based on the common temperature boundaries of mesophilic microorganisms. Identify this temperature range using the illustrated primer of the book “Biotechnology” from the year 2016 in the “basic Info material” folder.

Another task will be the design and cloning of the correct promoter sequence. A template will be given which serves as an aid for the creation of promoter sequences that meet the conditions for successfull promoter design. But you have to identify some hidden bases. Two important areas in the sequence to be identified are the recognition sequences that are responsible for gene expression. You can find out the optimal recognition sequences with the help of the paper “Bacterial sigma factors and anti-sigma factors” in the “basic Info material” folder. In order to perform a successful cloning, you have to design a suitable primer for each promoter. In addition, a melting temperature matching the primer sequence must be used. You can calculate the melting temperatures with the help of some formulas which you will find in the PDF-document called “Formulas for MW and TM calculation” in the “basic Info material” folder.