

# 1.1 Jupyter – Introduction

March 23, 2020

## 1 Introduction to Jupyter

This notebook is based on [Lectures on scientific computing with Python](#) by J.R. Johansson.

### 1.1 What is Jupyter Notebook?

The Jupyter Notebook is an **interactive computing environment** that enables users to create notebook documents that include live code, markdown text, plots, images and equations in LaTeX.

These documents provide a **complete and self-contained record of a computation** that can be converted to various formats and shared with others using email, [Dropbox](#), version control systems (like git/[GitHub](#)) or [nbviewer.jupyter.org](#) or [notebooks.azure.com](#).

The Jupyter Notebook combines three components: \* **Notebook editor**: an interactive application for writing and running code interactively and editing notebook documents. If you run Jupyter on desktop, you will be using Jupyter's web application, whereas right now you are using Juno as your notebook editor. \* **Kernels**: Separate processes started by Jupyter on your server, that runs users' code in a given language and returns output back to the notebook web application. The kernel also handles things like computations for interactive widgets, tab completion and introspection. \* **Notebook documents**: Self-contained documents that contain a representation of all content visible in the notebook editor, including inputs and outputs of the computations, markdown text, equations, images, and rich media representations of objects. Each notebook document has its own kernel.

### 1.2 Notebook editor

Notebook editor, be it Jupyter's web application running in a browser, or an app with embedded Jupyter like **Juno**, enables users to: \* **Edit code** with automatic syntax highlighting, indentation, and tab completion/introspection. \* **Run code**, with the results of computations attached to the code which generated them. \* See the results of computations with **rich media representations**, such as HTML, LaTeX, PNG, SVG, PDF, etc. \* Author **narrative text** using [Markdown](#) markup language. \* Include mathematical equations using **LaTeX syntax in Markdown**, which are rendered by [MathJax](#).

### 1.3 Kernels

Through Jupyter's kernel and messaging architecture, Jupyter Notebook allows code to be run in a range of different programming languages. For each notebook document that a user opens, the Jupyter's server application starts a kernel that runs the code for that notebook. Each kernel is

capable of running code in a single programming language and there are kernels available in [100+ languages](#), including **Python**, **Julia**, **R**, **Ruby**, **Haskell**, **Scala**, and many others.

The default kernel (and the only kernel currently available in Juno) runs Python code. The notebook provides a simple way for users to pick which of these kernels is used for a given notebook. Each of these kernels communicate with the notebook editor using JSON over ZeroMQ/WebSockets message protocol that is described [here](#). Most users don't need to know about these details, but it helps to understand that “kernels run code”.

## 1.4 Notebook documents

Notebook documents, or notebooks, contain the **inputs and outputs** of an interactive session as well as **narrative text** that accompanies the code but is not meant for execution. **Rich output** generated by running code, including HTML, images, video, and plots, is embedded in the notebook, which makes it a complete and self-contained record of a computation.

When you are using a notebook editor, notebook documents are just **files on your server's filesystem with a .ipynb extension**. This allows you to use familiar workflows for organizing your notebooks into folders and sharing them with others.

Notebooks consist of a **linear sequence of cells**. There are three basic cell types: \* **Code cells**: Input and output of live code that is run in the kernel. \* **Markdown cells**: Narrative text with embedded LaTeX equations. \* **Raw cells**: Unformatted text that is included, without modification, when notebooks are converted to different formats using `nbconvert`.

Internally, notebook documents are **JSON text files with binary data encoded in base64**. This allows them to be **read and manipulated programmatically** by any programming language. Because JSON is a text format, notebook documents are version control friendly.

**Notebooks can be exported** to different static formats including HTML, reStructuredText, LaTeX, PDF, and slide shows ([reveal.js](#)) using Jupyter's `nbconvert` utility.

Furthermore, any notebook document available from a **public URL on or GitHub can be shared** via [nbviewer](#). This service loads the notebook document from the URL and renders it as a static web page. The resulting web page may thus be shared with others **without their needing to install the Jupyter Notebook**.