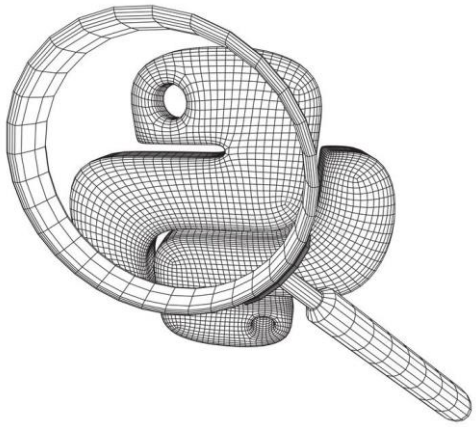
**Python Installation Guide for Beginners**

Dear students,

The **Sustainable Air Transport (SAT) MSc** profile will require you to develop a lot of computer code as part of different homework/assignments across multiple mandatory and elective courses. While your computer skills **will improve** throughout the MSc as you work on these assignments, an **initial level of proficiency is advised** to ease your transition from the BSc and make sure you can properly follow the courses and translate the theoretical concepts into proper working code.

As you stem from different backgrounds, come from different previous academic institutions and have different personal computer skills, we want to provide here a basic set of links with support material and tutorials to help you fill in some (potential) gaps that might hinder your learning experience.



The following document describes different aspects (from installation to tutorials related to some specific packages) of the **programming language Python**. Note that, while the use of Python is not an imposition, it is a **strong suggestion** for several reasons. For example, (1) it is the default programming language taught in the Aerospace Engineering BSc at TU Delft, (2) it is generally the programming language used by our staff members, (3) it is a high-level open-source programming language that supports, e.g., object-oriented programming, and (4) it is generally well-documented, either via "official documentation" or via the user community through resources such as *StackOverflow*.

As the following list assumes **little to no knowledge of Python**, it might be the case that these prerequisites are not relevant for most of the incoming SAT student cohort. If you are reading this and the following list covers topics and Python packages that you know well, this means you are off to a good start. If not, we suggest using the proposed material to fill in some gaps and ease the start of the SAT MSc track.

# 1. Python installation via Anaconda

### What is Anaconda?

Anaconda is a **free and open-source Python distribution** that can help you install hundreds of packages related to data science, scientific programming, development and more. Python is included in the Anaconda distribution. Note that it is **not an IDE** (like PyCharm) though it can be configured with most IDEs. Also note that the distribution **includes an IDE called Spyder** (reasonably good). It also comes with a platform-agnostic package manager called conda.

Note that while the proposed distribution is not the only solution to install python and its packages, using Anaconda has several advantages. Anaconda tries to be a "**Swiss army knife"**, as everything that can be manually installed using PIP, is also available in Anaconda and can be installed in Conda through a visual interface (GUI) or command line (see the command below).

```
# conda


conda install <package_name>
```

The good thing about Anaconda is that when you use the conda (either by command line or GUI) to install a python package, it usually pulls additional packages along with it, thus helping you to handle the package dependencies. Imagine you want to install **pandas** to handle a csv file. You would also need to install **numpy** which is a package that deals with large arrays. In Anaconda, this is transparent to you.

Downside of Anaconda? Conda distribution can occupy 2-4 GB of space very easily.

You can check these links to obtain the distribution and see how to install it:

https://www.anaconda.com/products/distribution

https://docs.anaconda.com/anaconda/install/

Regarding the IDEs, some students prefer IDEs such as PyCharm, but we do recommend the Spyder "default" option for beginners. You can find the application on the Anaconda Navigator at the Home tab.

# 2. Relevant packages to install

While Anaconda (or other distributions) comes with a large set of pre-installed packages, you might (and most likely will) need to install ad-hoc packages during your studies. When running a piece of code, you will immediately realize this, as Python will complain and say something like "**ModuleNotFoundError: No module named 'cartopy'**". There are generally two ways of installing new packages. If you are using Anaconda, you are generally advised to use the **conda** command from the command window (https://docs.conda.io/projects/conda/en/latest/user-guide/concepts/installing-with-conda.html). The reason why this command is advised is that it installs the package in the current active environment (a bit

more technical content here. Please check this link if interested: https://docs.python.org/3/tutorial/venv.html) and limits conflicts between environments. You can also use the GUI on the Anaconda Navigator, as explained previously.

Otherwise, the **pip** install option is the other default option, especially if you use other IDEs (https://packaging.python.org/en/latest/tutorials/installing-packages/).

**NumPy package**

NumPy is probably the Python package you will be using the most during your MSc studies. NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, providing a vast toolbox of mathematical functions to operate on these arrays. It also offers comprehensive mathematical functions, random number generators, linear algebra routines, Fourier transforms, and more. In a nutshell, it supports most of the basic and not so basic mathematical operators you will need.

Here is a good comprehensive tutorial:

https://www.youtube.com/watch?v=QUT1VHiLmmI

while here is a bit more advanced (and longer) tutorial:

https://www.youtube.com/watch?v=ZB7BZMhfPgk

**Pandas package**

Pandas is a Python package for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. Together with NumPy, it is probably the package our students use the most, as it can easily convert .txt, .csv, and .xlsx files into a Python data frame and offers a plethora of searching and filtering options to analyze datasets. As data handling and processing is a crucial part of our MSc program, mastering this package will severely enhance your data analysis skills.

Here is a good comprehensive tutorial:

https://www.youtube.com/watch?v=vmEHCJofslg

**Matplotlib**

Matplotlib is a Python package that allows plotting of data in many different forms, from scatter plots, to histograms and barplots, to object-oriented map projections, contour plots, and polar plots. It is the default to-go option for visualization purposes and, as such, it is a package that you will end up using it in basically all your courses.

Here is a fairly long, yet quite exhaustive, tutorial:

https://www.youtube.com/watch?v=wB9C0Mz9gSo

**Extras**

Some other packages you might need during your studies (either in some courses or when working on your MSc thesis) are listed here. Note that these packages are not considered prerequisites and hence are listed here, together with some tutorials, just for your (future) reference in case of need.

- **SciPy**: package used for scientific computing and technical computing (tutorial: https://www.youtube.com/watch?v=1TXNrW-pqo0).
- **scikit-learn**: a package that features various classification, regression and clustering algorithms, including support-vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy. As this package covers many different topics, each worthy its own tutorial, here is a quite long tutorial that covers most of the basic topics. It is divided topic-wise, hence it should be a good initial reference to get an overview: https://www.youtube.com/watch?v=pqNCD_5r0IU
- **datetime**: package that provides functions to work with dates, times and time intervals in terms of conversion between units, time zones, etc. Here is a good tutorial: https://www.youtube.com/watch?v=eirjjyP2qcQ

## Final remarks:

While the provided resources are in the form of (YouTube) tutorial videos, each of the listed packages has its documentation webpage, which is another to-go resource in case of need. For example, this is the official NumPy documentation webpage (https://numpy.org/doc/stable/). When it comes to more practical problems or you need to find implementation solutions, one of the best resources is *StackOverflow* (https://stackoverflow.com/) as, most likely, the implementation problem you are trying to solve has already been encountered (and hopefully solved!) by someone else.

Document edited by Alessandro Bombelli (a.bombelli@tudelft.nl). Other contributors: Bruno F. Santos (b.f.santos@tudelft.nl ), Marcia Lourenco Baptista (M.LBaptista@tudelft.nl).

Latest version: 12/07/2022