Local installation of Python and JupyterLab

Installing Python with Miniconda/Anaconda

Miniconda is a simple Python distribution that works on any operating system. Compared to Anaconda, you need to install all packages yourself. This is very helpful to learn how to manage Python environments, package versions etc. If you want the batteries-included Python installation, you can go with Anaconda instead.

Since the installation depends on your operating system, please find the instructions below:
https://www.anaconda.com/docs/getting-started/miniconda/install
https://www.anaconda.com/docs/getting-started/anaconda/install

After installing either distribution, open the `Anaconda Prompt` to install Python packages. You can either use `conda` or `pip` to install packages. The former is the package manager built into Miniconda/Anaconda, while the latter is the general package manager for Python.

Installing JupyterLab

JupyterLab is the local/single-user version of JupyterHub that you were using throughout the workshop. The user interface looks identical and you can continue using the same notebooks in JupyterLab. To install JupyterLab for your Python distribution, use *one* of the following commands in the Anaconda Prompt:

```
conda install -c conda-forge jupyterlab
pip install jupyterlab
```

After the installation, run the command `jupyter-lab` to start the local instance of JupyterLab. This should automatically open JupyterLab in your browser. See here for the detailed instructions.

Installing other packages

To install `numpy`, `matplotlib` and `pandas` with conda, run the following commands:

```
conda install numpy
conda install -c conda-forge matplotlib
conda install -c conda-forge pandas
```

To install these packages with pip, run the following commands:

```
pip install numpy
pip install matplotlib
pip install pandas
```

You can find the complete installation instructions in the respective documentations, see `numpy`, `matplotlib` and `pandas`. In general, any publicly-available package has installation instructions for either `conda` or `pip`.

Useful packages/resources

This is a list of other useful packages related to data analysis and data visualization in Python.

`seaborn` Advanced (and nicely looking) plots. The package is built on top of matplotlib and you can plot data directly from a pandas data frame.

`lmfit` Run fits on your data. The packages uses `scipy.optimize.curve_fit` internally, but it makes the handling of fit parameters and fit results a lot more convenient.

The Python Graph Gallery Many examples of plots made with `matplotlib`, `seaborn` etc. If you are looking for any inspiration on how to display your data, this is a great place to start.