

Spyder & Conda Environment Setup – Full Recap

This document summarizes the complete process used to fix Spyder, avoid breaking Jupyter, and create a clean and stable Conda environment setup. It includes the rationale and exact commands used, and can be kept as a reference or shared with a TA.

1. Initial Problem

Spyder failed to start correctly due to kernel and console errors caused by environment mixing and Windows Python path conflicts. Jupyter was already working and needed to remain untouched.

2. Final Design Decision

A professional separation of environments was adopted: base for Conda only, project8_env for Jupyter, and spyder_env for Spyder and scientific libraries.

3. Creating the Spyder Environment

```
conda create -n spyder_env python=3.10 spyder=5.5 spyder-kernels=2.5 -y
```

4. Installing Required Libraries

All scientific libraries were installed using conda-forge to ensure compatibility on Windows.

```
conda install -c conda-forge numpy=1.22.4
conda install -c conda-forge pandas=1.4.2
conda install -c conda-forge scipy=1.8.1
conda install -c conda-forge matplotlib=3.5.3
# OR
conda install -c conda-forge matplotlib=3.8
conda install -c conda-forge xarray=2022.6.0
conda install -c conda-forge geopandas=0.12.2
```

Matplotlib 3.5.3 was used for compatibility with legacy code, while Matplotlib 3.8 can be used when only public APIs are employed.

5. Launching Spyder

Spyder is launched from its dedicated environment using:

```
conda activate spyder_env
spyder
```

6. Final State

Spyder works correctly, all imports succeed, and Jupyter remains fully intact in project8_env.