Angiography + OCT Reconstruction in Python: Setting things up

Chris Mamon

1 Package/Environment Managers and Editors

Before we dive into the reconstruction we need to set up a python environment and editor. The package and virtual environment manager we have chosen is **Miniconda** and the editor is **Visual Studio Code** (VSC).

Miniconda is a free minimal installer for conda. We will use it to manage python packages and environments. If you are unfamiliar with these two terms:

- Packages: Are hierarchical structures containing python programs.
- Virtual Environments: Isolate python dependencies from ones another, allowing developers to easily separate projects on the same device.

Figure 1: Miniconda download and documentation page `https://docs.conda.io/en/latest/miniconda.html`

Visual Studio Code is a source-code editor developed by Microsoft for Windows, Linux and macOS. It includes support for debugging, embedded Git control and GitHub, syntax highlighting, intelligent code completion, snippets, and code refactoring. It is my personal editor of choice due to its light weight nature, high customizability and powerful debugging.

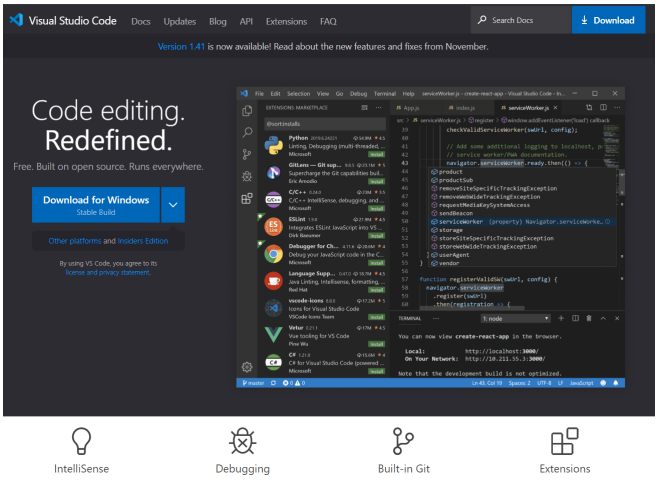

Figure 2: Visual studio code homepage `https://code.visualstudio.com/`

2 Github

Github is a Git repository hosting service. Git is a distributed version-control system for tracking changes in source code during software development. The source code for this project hosted in github and can be downloaded directly from `https://github.com/hj40/pymethods.git`.

3 Setting up a Miniconda Environment in Visual Studio Code

To use our miniconda extension in VSC we first need to install the Anaconda Extension Pack. This can be done directly through the VSC extensions tab, figure 3

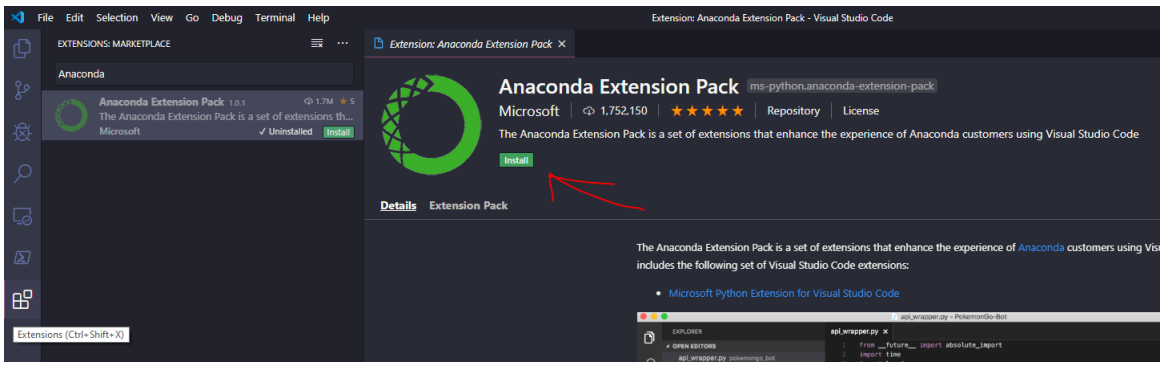

Figure 3: VSC Extension market place press the green install button to install the Anaconda Extension Pack

Let us now open us a new integrated terminal in VSC.

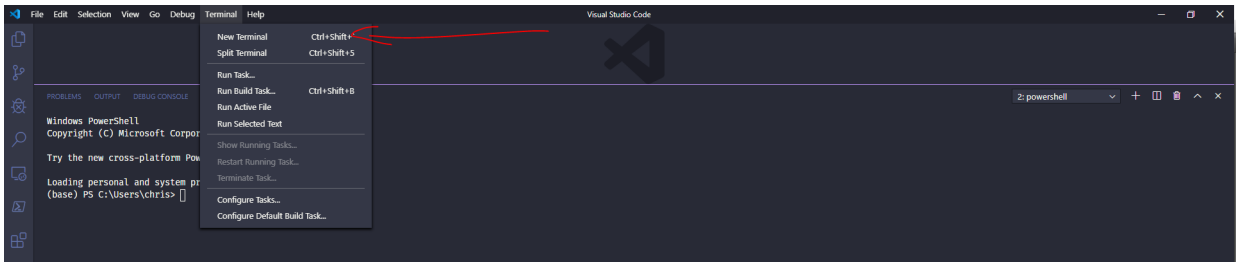

Figure 4: Opening a new terminal in VSC. The integrated terminal has popped up bellow. More information about the terminal can be found at `https://code.visualstudio.com/docs/editor/integrated-terminal`

From the terminal we can create and activate miniconda. If using powershell, i.e. using windows we must first call

```
conda init powershell
```

To create a miniconda environment called *myenv* we use the following command.

```
conda create --name myenv
```

Once the environment is created we can activate it using ,

```
conda activate myenv
```

4 Environment Variable Definitions

Let us say we are developing on multiple packages and want to easily mingle them together. For example we have the following packages,

```
/
├── package A
│   └── modules A
├── package B
│   └── modules B
└── package C
    └── modules C
```

We must include these in the *PYTHONPATH* environment variable. Though this can be done within a terminal, the specified changes are transient. To make them persist we can modify our *profile/rc* files.

4.1 Windows Powershell

In windows Powershell this is managed from the %PROFILE% file. In VSC we can run,

```
code $PROFILE$
```

This will open up the file within VSC, and we can add the following lines to the file,

```
$env:PYTHONPATH = "$env:PYTHONPATH;<path to package A>;<path to package B>;..."
```

4.2 Mac Bash

In MAC OSX Bash our profile is managed in the bash_profile file located from within our $USER directory. To open the file we can run,

```
code $USER/.bash_profile
```

Now to extend out PYTHONPATH we add the following lines

```
export PYTHONPATH="${PYTHONPATH}:<path to package A>:<path to package B:..."
```

Perform this for the repository downloaded from github

5 Setting up a folder in VSC

Now we can open up the package folder in VSC to the repository downloaded from github, figure 5. This allows us to view the directory tree structure in visual studio and apply linting and debugging tools.

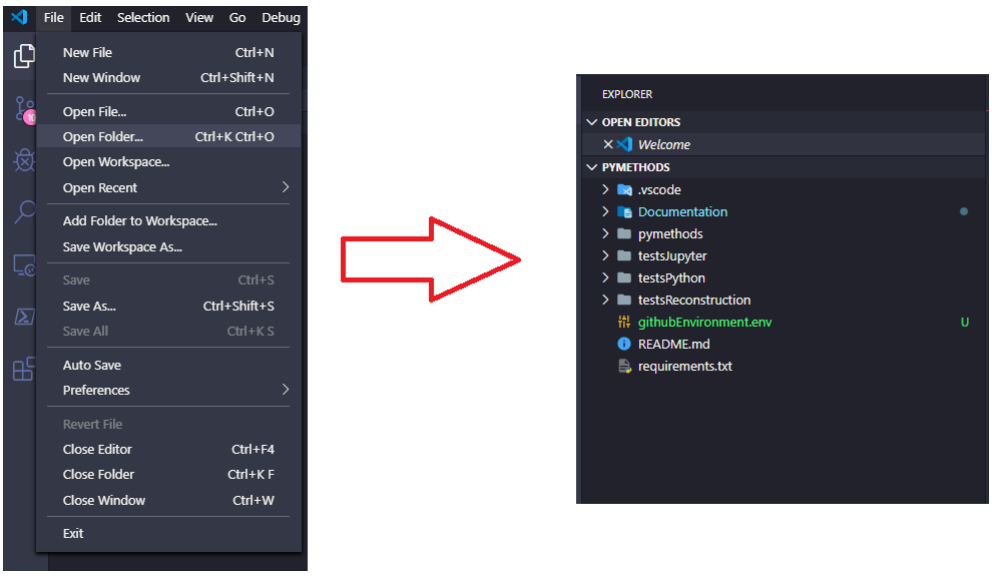

Figure 5: Opening a folder in visual studio code

Initially our conda environment is might not be selected. To select our desired python interpreter (i.e. environment) we can open a command panel with `ctr+shift+p` and by searching for `python interpreter`, 6, we can view a list of our available python interpreters, figure 7.

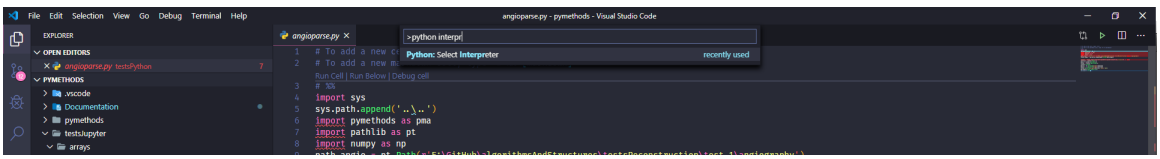

Figure 6: Finding the select interpreter method from the command panel

we can also click the interpreter button at the bottom left of the screen 7,

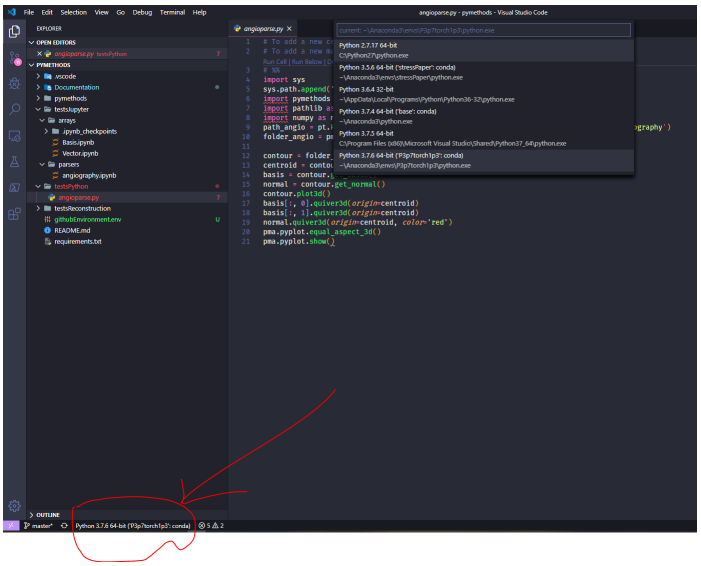

Figure 7: Interpreter selection after running select interpreter from the command panel. This panel can also be opened interactively by pressing the button circled in red

6 Installing Requirements

once the interpreter has been selected the python packages required for the Angiography + OCT Reconstruction package can be installed quickly via the pip package installer. Run the following command from the terminal,

```
python -m pip install -r <path>/requirements.txt
```