Working with conda

I would suggest conda as package management when working with python. For compiling software the system-installation of python should be used. This prevents mismatches between package versions, since programmers know what versions e.g. come with a Linux distro. Both installations can exists side-to-side with each other. You can activate your conda environment after installation as follows:

```
$ conda activate
(base) $
```

This will set all necessary paths. While inside of an conda environment it can be closed with

```
(base) $ conda deactivate
$
```

Furthermore, virtual environment are a useful tool to keep only the dependencies for a project that you need. Keeping the environment clean and easier to set up again. To create a new virt. env. use the following command:

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

myenv can be replaced by the desired name. You can enter a specific env with

```
$ conda activate myenv
(myenv) $
```

Packages can be installed with

```
(myenv) $ conda install ...
```

By default conda will install the package in the active environment. With the -n flag a user can install packages in the environment without activating it. Same goes for commands like conda update to update packages.

To show all avlaible environments one can use

```
$ conda env list
```

To see all installed packages use

```
(myenv) $ conda list
```

Packages in conda come in channels from which they can installed. The -c flag during installation can be used to mark a specific channel. A popular channel for example is conda-forge. There one finds many useful packages like pyroot and conda-related packages e.g. There can be a list of different channels, these are usualy given in decreasing priotiy. conda will try to install a package and all dependencies from the channel with highest priority first. To list all active channels use

```
(myenv) $ conda config --show channels
```

To add the channel conda-forge in the virtual environment use

```
(myenv) $ conda config --add channels conda-forge
```

I would recommend adding the following channel as highest priority to get the official pytorch releases

```
(myenv) $ conda config --add channels pytorch
```

Jupyter Notebook

Jupyter notebooks are a great resource for testing. Data has to be loaded only once and afterwards one can at leisure adjust plots or test the functionality of a function. When working with virtual environments one could install the notebook in everywhere and use it, but it is possible to install and setup the notebook just once in the base environment and afterwards use kernels from every virtual environment.

Install conda and activate your base environment. Then use the following to install jupyter notebook

```
(base) $ conda install -c conda-forge notebook
(base) $ conda install -c conda-forge nb_conda_kernels
```

Additionally one can install Notebook extensions. These can be useful but are not necessary

```
(base) $ conda install -c conda-forge jupyter_contrib_nbextensions
```

To be able to use the virtual environment from the Jupyter Notebook in the base environment one need to install one additional package. Activate the virtual environment and use

```
(myenv) $ conda install ipykernel
```

Now one should be able to choose kernel from the virtual environment for a notebook. Also when using new to create a new notebook one should be able to choose between the different environments. The package ipykernel has to be installed in every environment that the Jupyter Notebook from the base environment needs to access.