Step-by-Step Guide to Set Up Python, NumPy, SciPy, and Jupyter on a Windows Machine Using Miniforge

---

Introduction

Python is a powerful programming language that is widely used in various fields, from web development to scientific computing. **NumPy** and **SciPy** are essential libraries for numerical and scientific computing, providing tools to perform complex mathematical operations efficiently. **Jupyter Notebook** is an interactive web application that allows you to create and share documents containing live code, equations, visualizations, and narrative text.

- Python documentation
- NumPy documentation
- SciPy documentation
- Jupyter documentation

Step-by-Step Instructions

Install Git for Windows

1. If you haven't installed Git yet, download it from the Git for Windows website
2. Run the installer and follow the on-screen instructions. During the installation process, ensure that **"Open Git BASH Here"** option is selected.

---

1. **Open the Git BASH Terminal**
   A. Click on the **Start** menu or press the **Windows** key.
   B. Type **"Git BASH"** in the search bar.
   C. Click on the **Git BASH** app from the search results to open the terminal.

---

2. **Install Miniforge**

   Miniforge is a community-driven, cross-platform package manager built on top of conda.

   A. Download the Miniforge installer for macOS with Intel architecture:

```
curl -L -O https://github.com/conda-forge/miniforge/
releases/latest/download/Miniforge3-Windows-x86_64.exe
```

B. Run the installer:

```
./Miniforge3-Windows-x86_64.exe /S /D=C:\Miniforge3
```

Here, /S is for silent mode and /D specifies the installation directory. Adjust the installation directory as needed.

C. Follow the on-screen instructions if prompted. Otherwise, the silent installation will proceed with default settings.

D. After installation, close and reopen Git BASH to ensure the changes take effect.

E. Verify the installation by checking the conda version:

```
conda --version
```

---

3. **Create a Virtual Environment**

A **virtual environment** is a self-contained directory that contains a Python installation built around a specific version of python with additional packages. **Virtual environments** allow you to have multiple builds of Python for different purposes.

A. Verify Miniforge is installed correctly by checking the version:

```
conda --version
```

B. Create a new virtual environment named 'phys3510' with Python 3.9:

```
conda create --name phys3510 python=3.9
```

C. Activate the virtual environment:

```
conda activate phys3510
```

D. Verify the virtual environment is active. You should see **`(phys3510)`** at the beginning of your terminal prompt.

---

4. **Install NumPy, SciPy, and Jupyter**

A. Install the necessary packages from the conda-forge channel:

```
conda install -c conda-forge numpy scipy jupyter
```

B. Verify the installations:

```
python --version

python -c "import numpy; print(numpy.__version__)"

python -c "import scipy; print(scipy.__version__)"

jupyter --version
```

---

5. **Create or Navigate to a Directory for Your Notebooks**

   Aside: Basic Terminal Commands

- **`pwd`** : Shows the current directory you are in.
- **`ls`** : Lists the files and folders in the current directory.
  - **`ls -l`** : tells `ls` to show the files and folders in a list format
- **`cd <directory_name>`** : Changes the current directory to the specified directory.
  - **`cd ..`** : Moves up one directory level.
  - **`cd ~`** : Moves you to your home directory.
- **`mkdir <directory_name>`** : Creates a new directory with the specified name.

   Tip: When working in the terminal, its normal to use **`pwd`** , **`ls -l`** , and **`cd ..`** to navigate. When you get lost, you can use **`cd ~`** to reach the home directory and start over.

   Back to the Task: Steps to Create a New Directory for your Jupyter Notebooks

1. Check where you are with `pwd` .
2. List the files and folders in your current directory with `ls` or `ls -l` .
3. Navigate to where you want to store your Notebooks with `cd <directory_name>` .
4. Create a new directory (folder):

   ```
   mkdir JupyterNotebooks
   ```

5. Navigate to your JupyterNotebooks directory:

   ```
   cd JupyterNotebooks
   ```

---

6. **Launch Jupyter Notebook**
   A. Start Jupyter Notebook:

      ```
      jupyter notebook
      ```

   B. A new tab should open in your default web browser with the Jupyter Notebook interface.

---

7. **Work in Jupyter Notebook**
   A. Create a new Python notebook by clicking **File > New > Notebook**.
   B. You will be prompted to choose a kernal. A kernel is the computational engine that runs your code. **Python 3** should be the default.

---

8. **Close Jupyter Notebook and Deactivate the Environment**
   A. To stop the Jupyter Notebook, go back to the Terminal where Jupyter was started and press **`Cmd+C`**.

   B. Confirm the shotdown by typing `y` when prompted and press Enter/Return.

   C. Deactivate the virtual environment:

      ```
      conda deactivate
      ```

Typical Usage Workflow

1. Open your Terminal and activate your virtual environment with `conda activate phys3510`.
2. Navigate to your JupyterNotebooks directory/folder with `cd` and `ls`. For example, `cd JupyterNotebooks`.
3. Launch Jupyter Notebook to start working on your project with `jupyter notebook`.
4. Create or open a Jupyter notebook and write your code.
5. Save your work regularly by clicking the save icon or pressing **`Cmd+S`**.
6. Whn you are done, close the **Jupyter Notebook** by going to the Terminal and pressing **`Cmd+C`** and confirming with `y`.
7. Deactivate the virtual environment with `conda deactivate`.

Additional Information

Some useful links:

- Navigating the Terminal.
- More about Jupyter Notebooks.
- MarkDown basics. This is how we keep notes and organize our Jupyter Notebooks.

You can remove a virtual environment with the following command: `conda remove --name phys3510 --all`

In [ ]: