

How to Set up VS code to work remotely with ODHAR

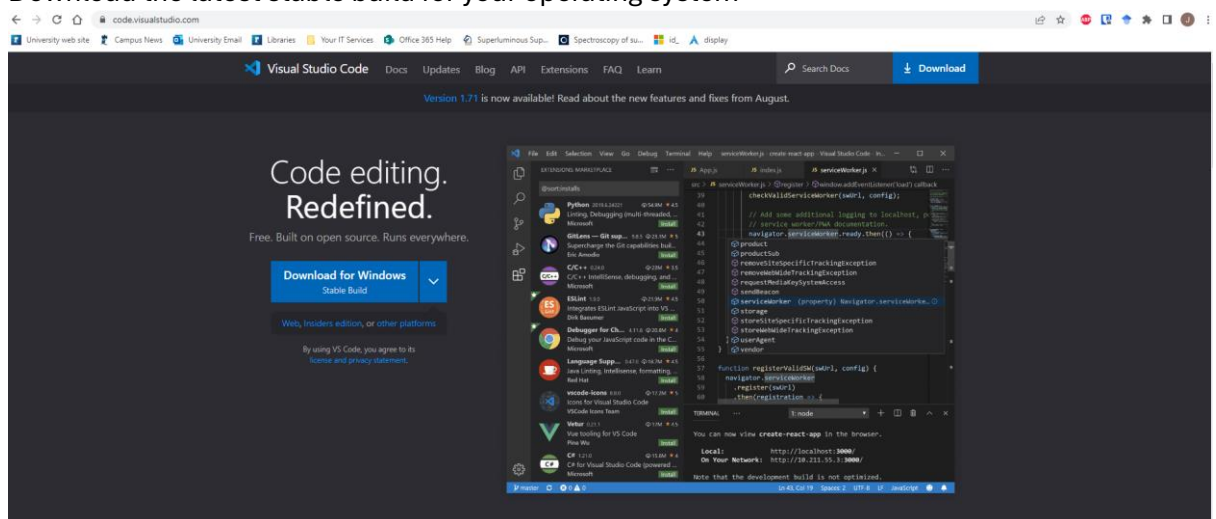
Having an efficient and usable programming environment provides a smooth workflow for you when you have larger issues to worry about. These instructions were created with the intent to show you how to set up one IDE (VS code) and connect it to our local computing cluster Odhar. By doing this you will be able to create fast code without needing powerful hardware and will be able to access your code anywhere (including your home!).

Step 0: Login to the VPN

If you are working remotely, such as at home, login to the VPN before starting the next steps. The VPN we use is FortiClient and requires Microsoft authenticator on your phone to function.

Step 1: Install VS code

1. Navigate to the VS code home page
2. Download the latest stable build for your operating system

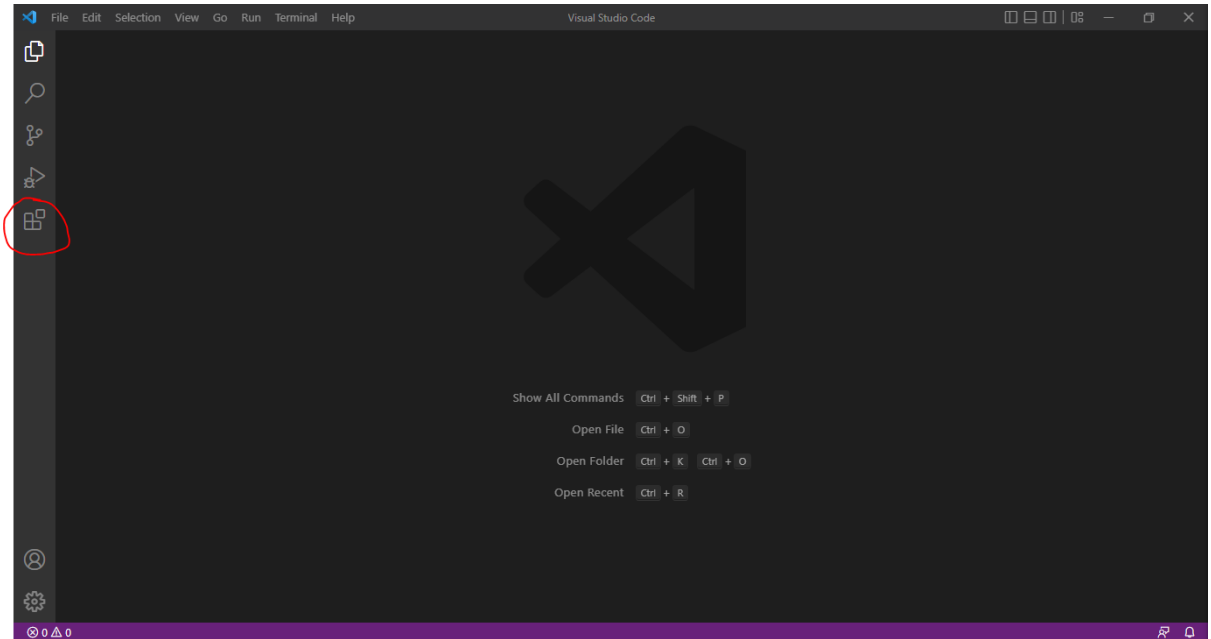


3. All the default options on the installer are fine, when finished launch VS code

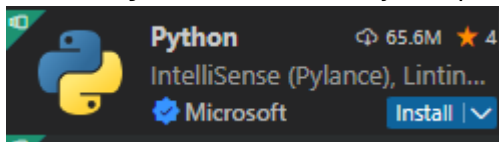
Step 2: Install required extensions

These instructions will assume you wish to program in python. However, VS code supports most languages you will encounter, with guides to setup and run them available online.

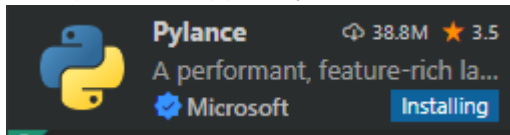
1. Navigate to the “Extensions” tab on the left toolbar



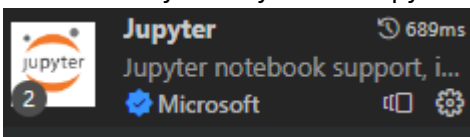
2. Search “Python”, install the Python plugin



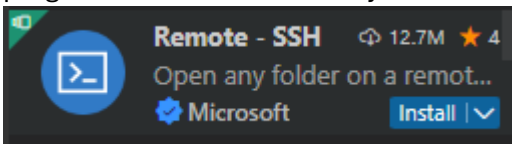
3. Search “Pylance”, install the Pylance plugin (this may install for you automatically when you install python)



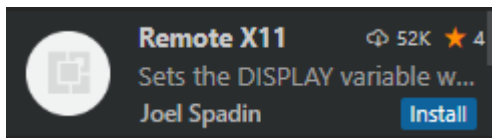
4. Search “Jupyter”, install the Jupyter plugin. This plugin allows you to program using Jupyter notebooks, a powerful tool for prototyping code. (this may install for you automatically when you install python)



5. Search “Remote-SSH”, Install the Remote – SSH plugin. This plugin allows you to program and run code directly on remote computers (such as Odhar) from your laptop



6. Search “Remote X11”, Install the Remote X11 plugin. This plugin allows VS code to forward any graphs you generate remotely to your laptop, as if you had generated the plot locally. You can program and run code without Remote X11, however you will have to manually save and download plots.

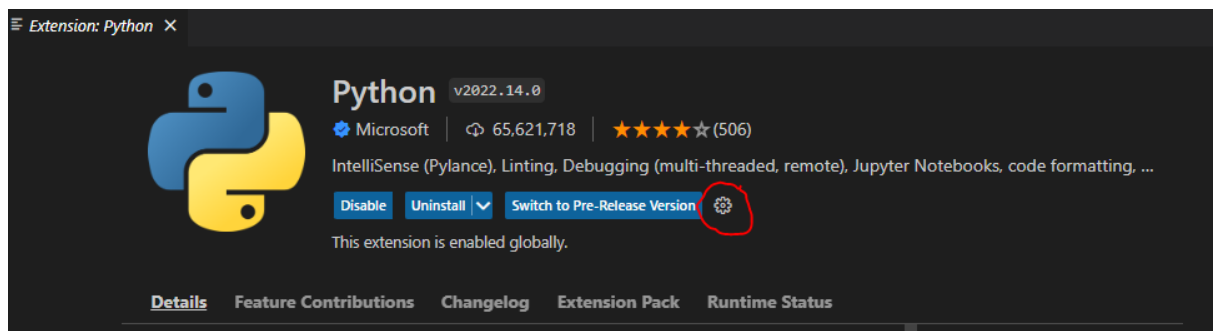


You should now have all the extensions needed for programming in python. VS code has thousands of useful extensions however, it is worth exploring some of potential extensions out there that may significantly speed up your programming speed.

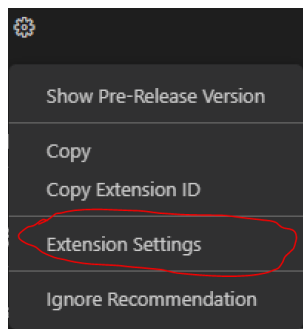
Step 3 (Optional): Editing any personal settings

Some people have a preferred python version (e.g. those still stuck on python 2). Additionally, you may have some preferences on how python code is interpreted. You can tell VS code which version you wish to use, and access any other python settings by:

1. Open the extensions tab
2. Search “Python”
3. Click the Python extension
4. Click the Gear icon



5. Click “Extension Settings”

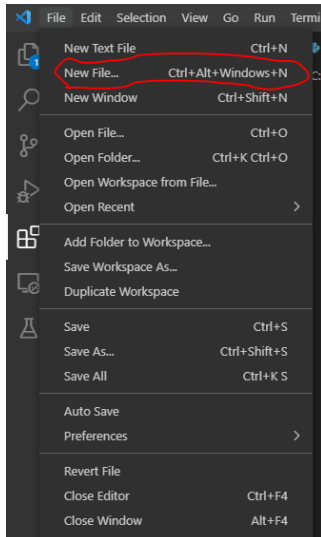


6. If you have a preferred interpreter, select “Default Interpreter Path” and navigate to the version of python you wish to use (e.g. For python version 3.10 on windows your python interpreters are most likely stored under “C:\Program Files\Python310 “ or “C:\Users\USER_NAME\AppData\Local\Programs\Python\Python310”)

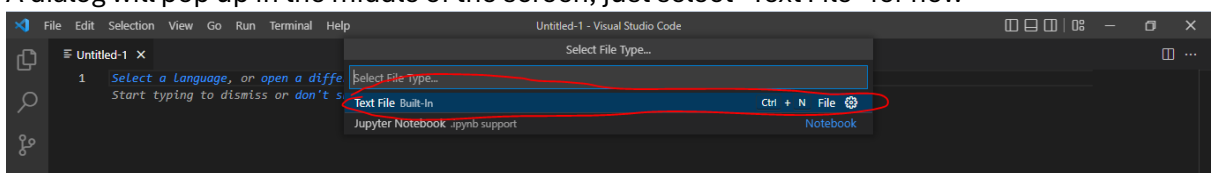
Step 4: Testing everything is installed correctly

While we have not set up remote programming yet, it would be prudent to test if python is working locally first before moving on.

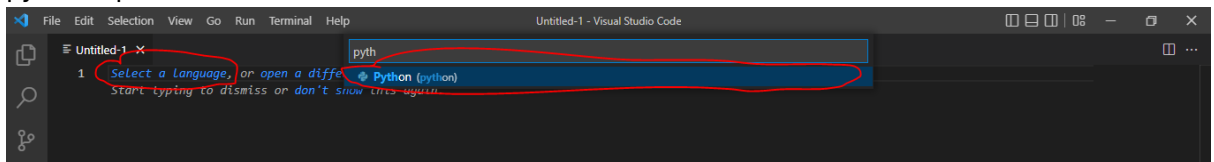
1. Create a new file in VS code



2. A dialog will pop up in the middle of the screen, just select “Text File” for now



3. Click “Select a language”, on the dialog that comes up, search python and select the python option

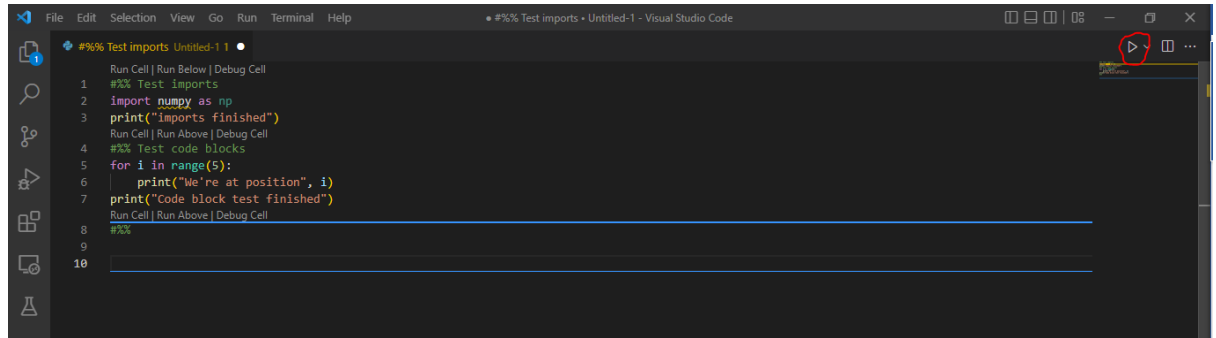


4. Type the code snippet shown below, this code provides a quick test of both Python and Jupyter

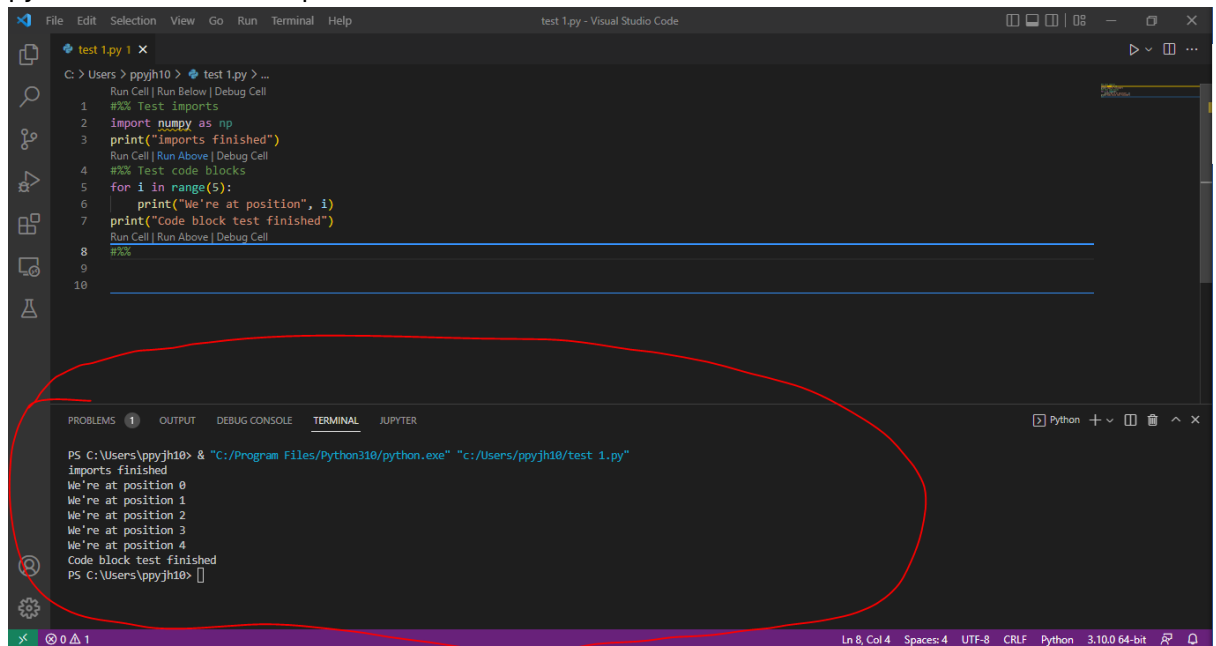
```
#%% Test imports
import numpy as np
print("imports finished")
#%% Test code blocks
for i in range(5):
    print("We're at position", i)
print("Code block test finished")
#%%
```

5. Once the code is written, save the file somewhere in your computer.

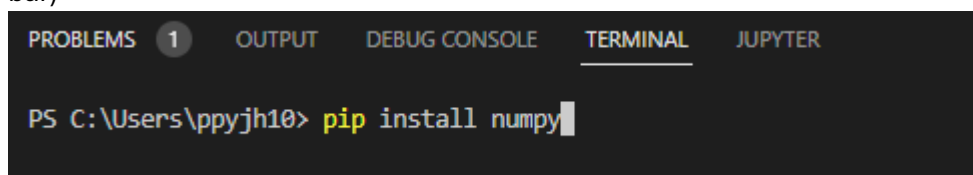
6. Press the “run” button on the top right (this looks like a play symbol)



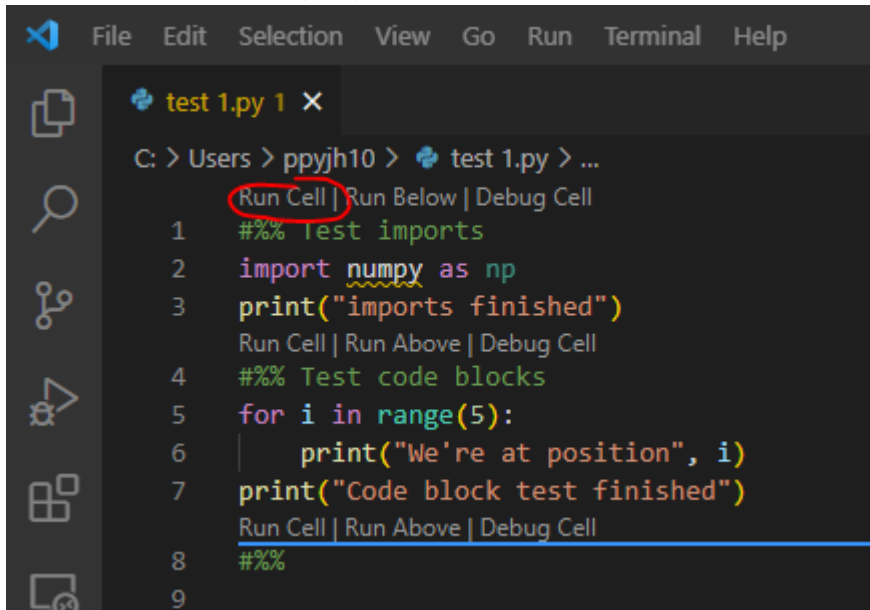
7. A terminal should pop up at the bottom of your VS code window, here the output of your python code should be printed



8. You may get an error saying “numpy is not installed” if this is the case, run the command “pip install numpy” in the terminal (if you don’t have the terminal open you can use the shortcut **Ctrl + Shift + `** to open it again, alternatively, you can drag up from the purple bar)



9. To test Jupyter, instead of running the code using the run button press the “Run Cell” command that shows up in your code.

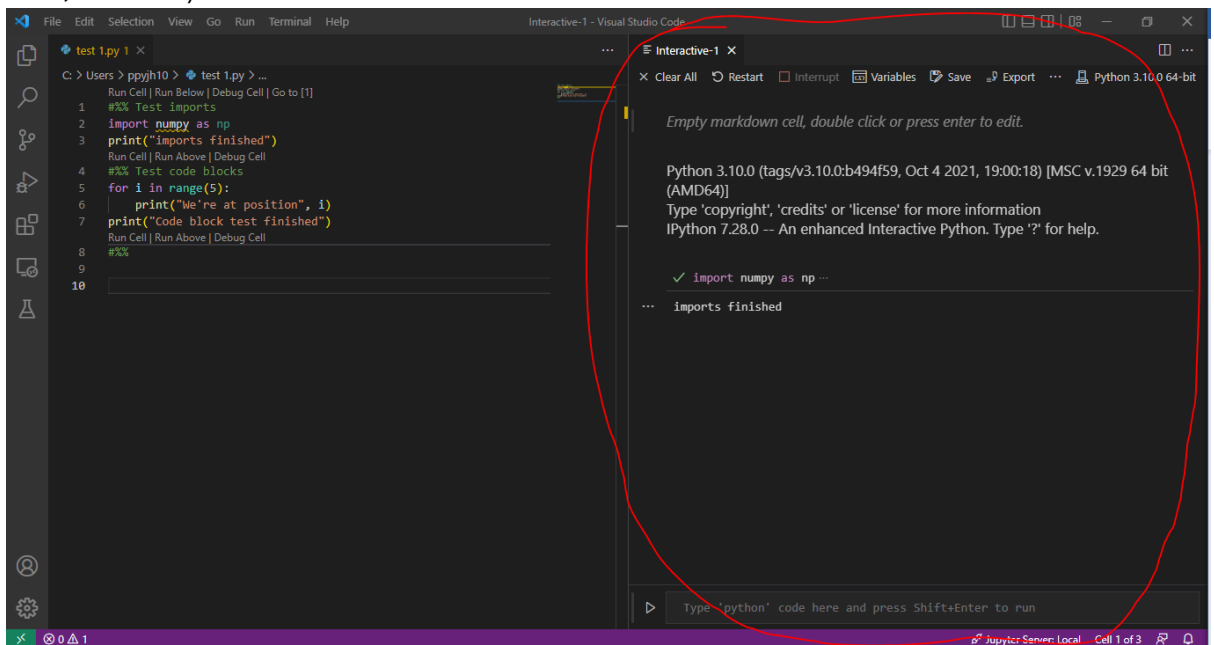


The screenshot shows a code editor window with a file named 'test 1.py'. The code is as follows:

```
C: > Users > ppyjh10 > test 1.py > ...  
1  Run Cell | Run Below | Debug Cell  
2  #%% Test imports  
3  import numpy as np  
4  print("imports finished")  
5  Run Cell | Run Above | Debug Cell  
6  #%% Test code blocks  
7  for i in range(5):  
8      print("We're at position", i)  
9  print("Code block test finished")  
10 Run Cell | Run Above | Debug Cell  
11 #%%  
12
```

The 'Run Cell' button for the first code block is circled in red.

10. This will open a new window with a stripped down Jupyter notebook session on your right. Jupyter notebook sessions allow quick prototyping of code by remembering what has been run previously. This allows you to run the parts of your code that you know work just once, while tweaking and rerunning the small part of your code you are trying to get working (this is especially useful in physics as we often deal with very large, and slow, datasets).

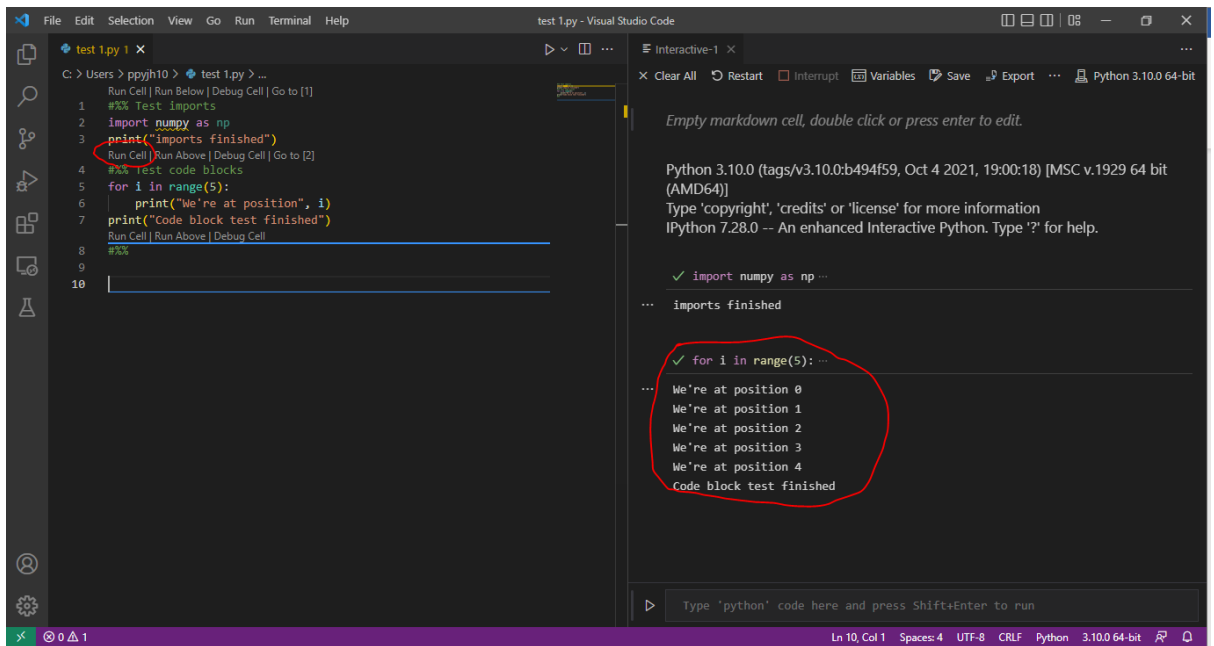


The screenshot shows a Jupyter notebook session in a window titled 'Interactive-1 - Visual Studio Code'. The notebook contains the following output:

```
Python 3.10.0 (tags/v3.10.0:b494f59, Oct 4 2021, 19:00:18) [MSC v.1929 64 bit (AMD64)]  
Type 'copyright', 'credits' or 'license' for more information  
IPython 7.28.0 -- An enhanced Interactive Python. Type '?' for help.  
  
✓ import numpy as np ...  
... imports finished
```

The notebook window is circled in red.

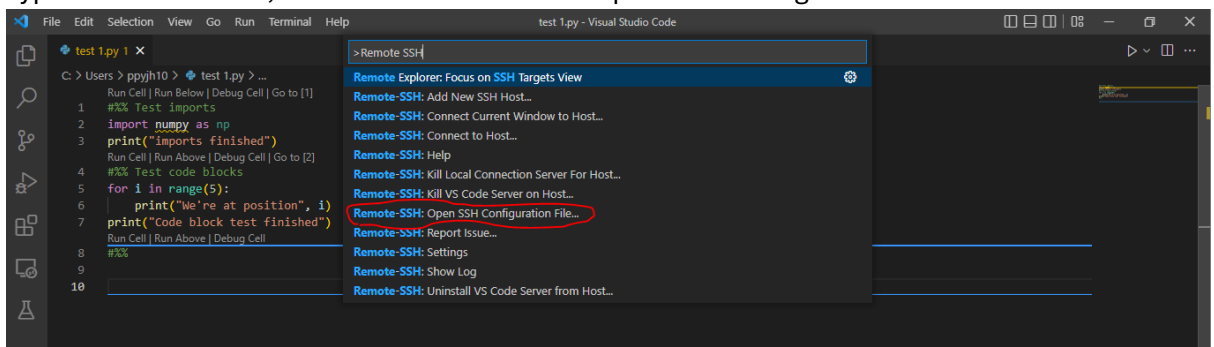
11. You will notice the Jupyter notebook code has not outputted our for loop, it only imported “numpy”. This is because our test code is split into two **code blocks**. You can create code blocks in normal python code by using the symbols “#%%”, normal python code treats these symbols as a standard comment and ignores them, but Jupyter interprets these as you breaking up your code into individual blocks. To run the rest of our code, press the second “Run Cell” button



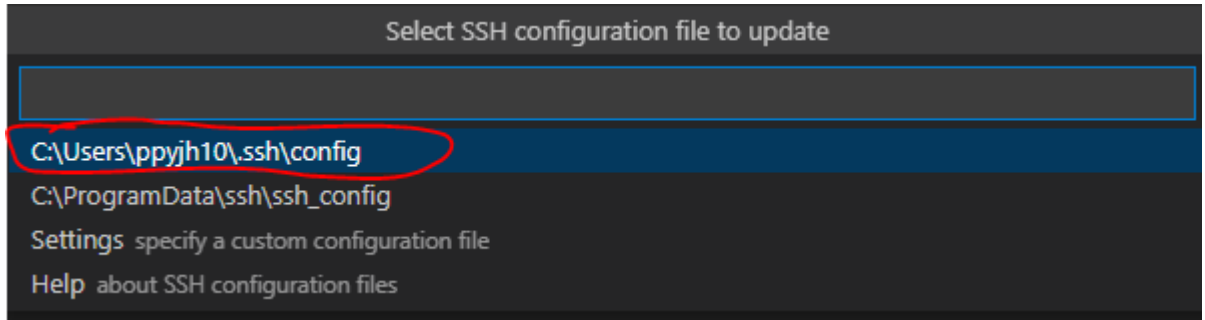
Step 5 : Setting up an SSH Host (such as ODHAR)

SSH stands for “Secure Shell”. Traditionally, it is a way of accessing unix/linux systems remotely. However, it can also be used to transmit data (such as programming commands) to a remote computer. This is useful for us Physicists as we often deal with massive datasets that would be too large to store locally and take too much time to process on a laptop. By using SSH we can run our code on a high performance computing cluster without needing to leave our beds.

1. In VS code, Press **Ctrl + Shift + P** to open VS codes “Command **P**alette”. This is a versatile search in VS code that can bring you pretty much anything if you know how to use it (for example, if you type “Select Interpreter” you can change your python interpreter on the fly from here, instead of going through the settings).
2. Type “**Remote SSH**”, and click “Remote-SSH: Open SSH Configuration File”



- Click the first option (something like “C:\User\YourName\.ssh\config”)



- This will open up your options file, add the lines (replacing “ppyjh10” with whatever your username is):

```
Host captain.physics.nottingham.ac.uk
HostName captain.physics.nottingham.ac.uk
User ppyjh10
ForwardAgent yes
ForwardX11 yes
ForwardX11Trusted yes
```

ForwardX11 and ForwardX11Trusted allow graphs and images to be sent via your SSH connection (this is equivalent to the -X and -Y flags in traditional SSH commands).

ForwardAgent allows you to sign in without needing to input your password each time (which we will set up fully in a later step) by using a randomly generated key instead.

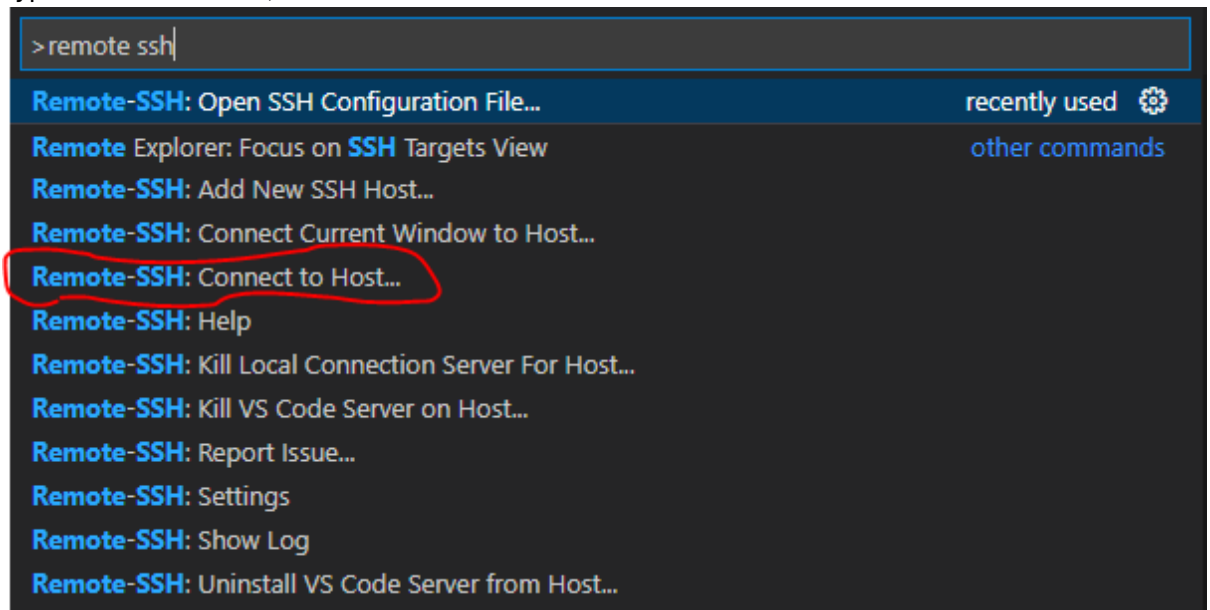
You have now set up your SSH host. If you want to add new hosts later, simply repeat these steps, adding any new lines to the config. For example, I often access a computer known and “Pleiades”, my config file looks like this:

```
C: > Users > ppyjh10 > .ssh > config
1 Host odhar.physics.nottingham.ac.uk
2 HostName odhar.physics.nottingham.ac.uk
3 User ppyjh10
4 ForwardAgent yes
5 ForwardX11 yes
6 ForwardX11Trusted yes
7
8 Host pleiades.physics.nottingham.ac.uk
9 HostName pleiades.physics.nottingham.ac.uk
10 User ppyjh10
11 Port 22
12 IdentityFile C:/Users/ppyjh10/.ssh/id_rsa
13
14 Host captain.nottingham.ac.uk
15 HostName captain.nottingham.ac.uk
16 User ppyjh10
17
```

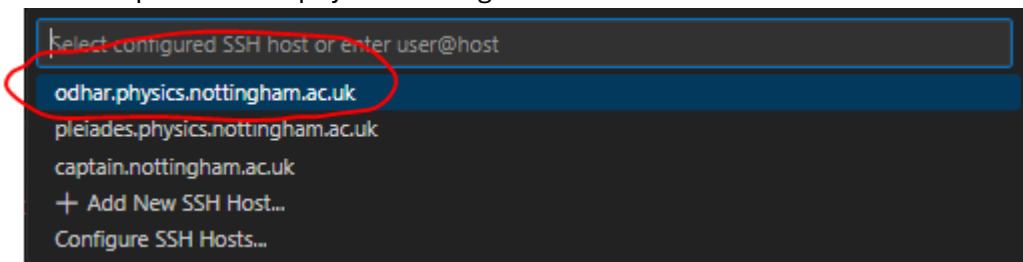
Step 6 : Connecting to an SSH Host (such as ODHAR)

Now that SSH has been fully set up in VS code, we can try connecting to ODHAR remotely!

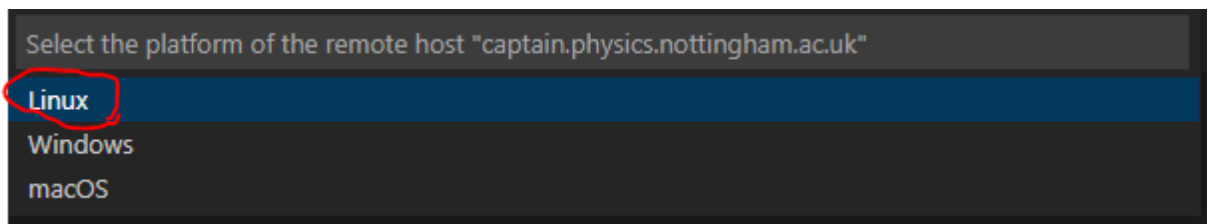
1. Open the “Command Pallet” (**Ctrl + Shift + P**)
2. Type “**Remote-SSH**”, click “Remote-SSH: Connect to Host...”



3. Click the option “odhar.physics.nottingham.ac.uk”

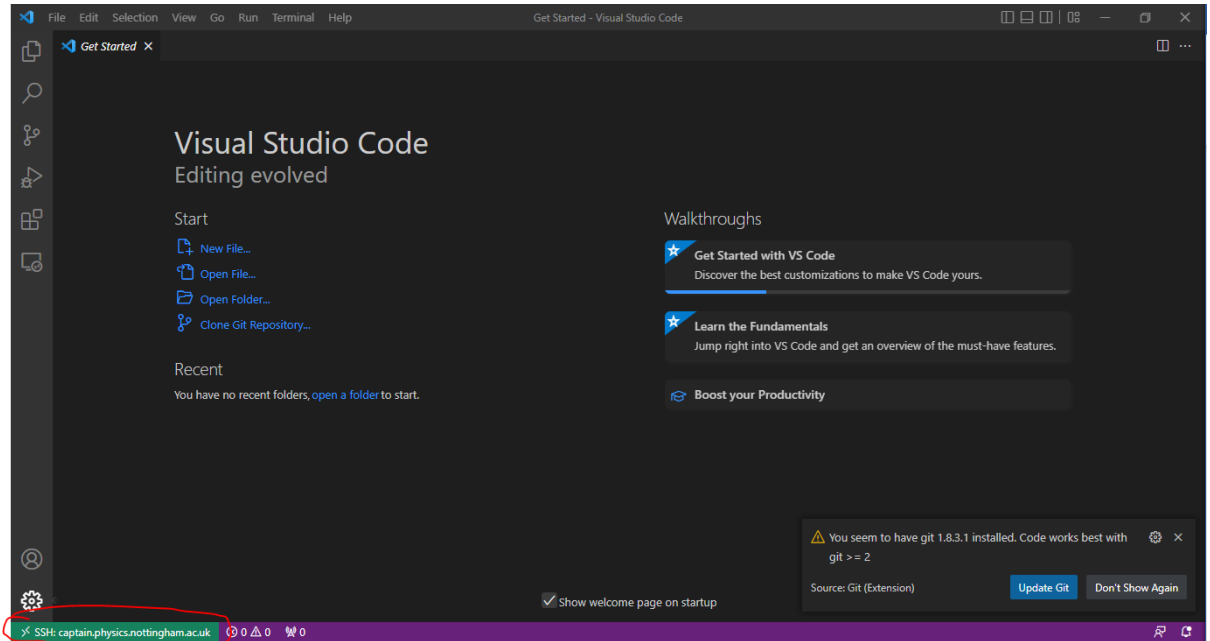


4. A new window should have opened, after a few seconds a popup in the middle of the window should appear asking what platform the remote host uses. Odhar is Unix based so we select Linux

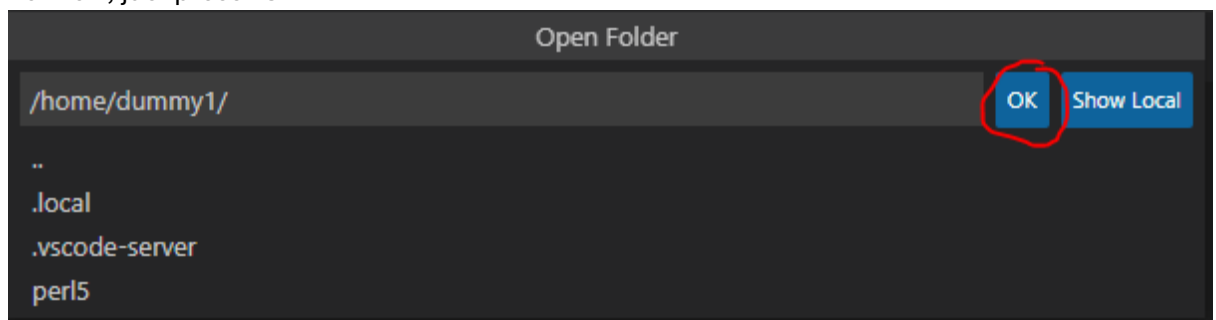


5. The popup will now ask you to enter your password, this will be whatever your uni password is, if you ever need this changed or reset ask Phil! Once entered, you will know if you are connected by looking at the bottom left of your window, in the green box it should say “**odhar.physics.nottingham.ac.uk**” (for future reference, you can click this green box to immediately open the “Command Pallet” with all the Open – SSH options

pre-selected)

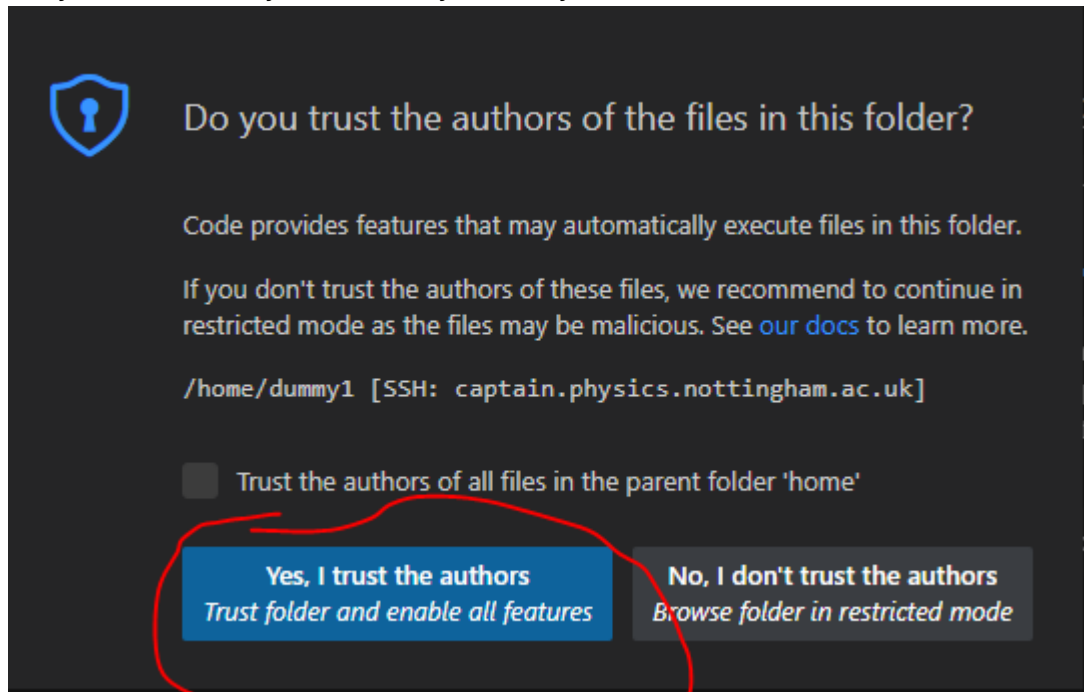


6. You can open your home directory by clicking “Open Folder”, this will open a popup asking which folder you wish to open. It will default to your home directory inside Odhar. For now, just press “OK”

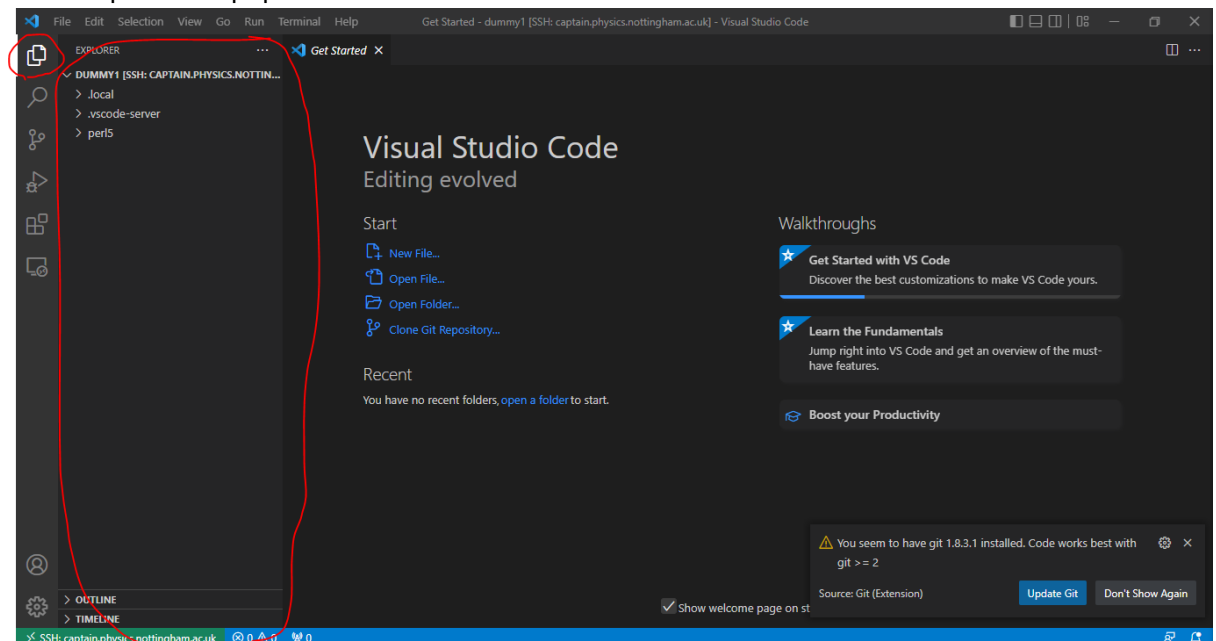


7. This will refresh the window, and you will have to enter your password again (we will fix this in a later step). It may now ask you if you trust the authors of the files in this folder,

ask yourself truthfully “do I trust myself?”, if yes then select “Yes I trust the authors”

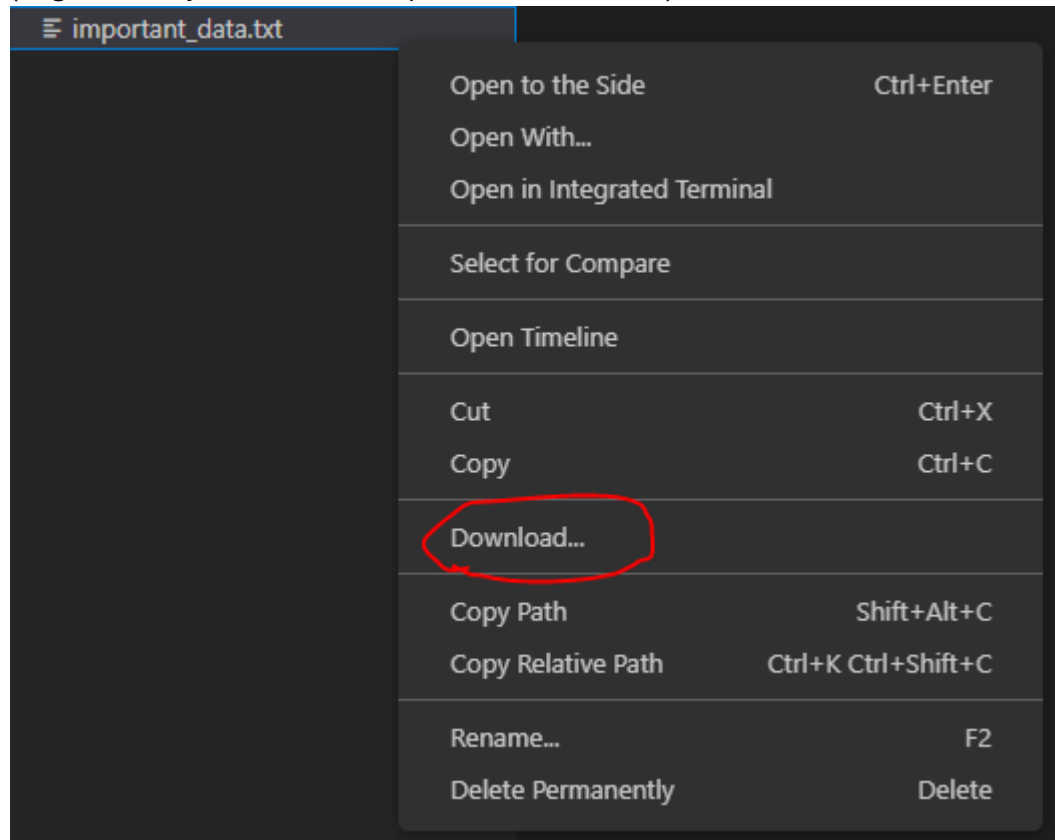


8. You now will have a panel open on the right of your VS code window, this will show your folders and files, if you ever close this you can reopen it by clicking the icon that looks like two pieces of paper



9. Here you can do anything that you can usually do to a file structure, such as: create, rename, move, or delete files and folders. You can also download files and folders to your home machine by clicking “Download”. To upload files and folders to your directory in Odhar, simply drag and drop the files into this explorer window, and they will begin to upload

(large files may take a while to upload and download)

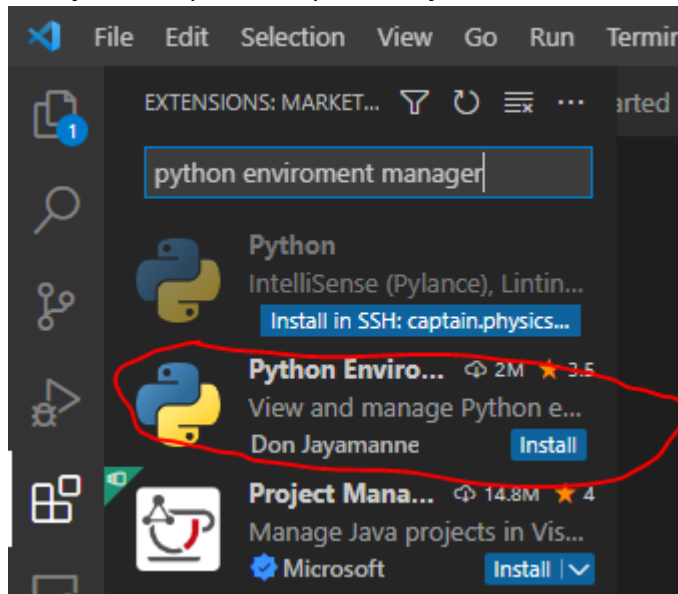


Step 7: Setting up your SSH environment

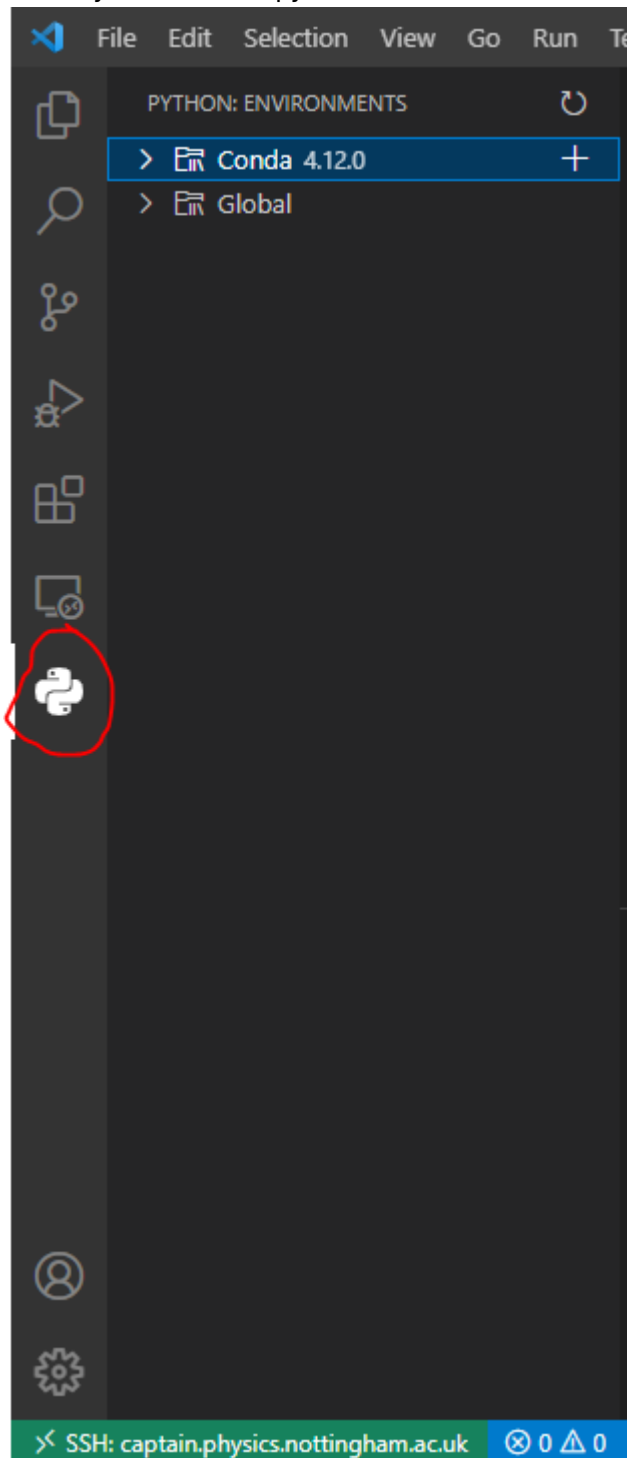
You have now connected to ODHAR, but python has not been set up here for you yet. As ODHAR is a shared machine, we can't just install everything we want to use globally like we might do on our own computer. To make sure all users can use what packages they want, we use part of the python IDE known as “**Anaconda**”, which contains an environment manager under the name “**Conda**”. This allows you to install modules and packages to your local directory, which prevents you accidentally ruining someone else's code with an incompatible package.

1. If you are not currently, connect to ODHAR in VS code (either open the Command Palette and search Open SSH and find the “Connect to Host” option, or press the green “X” at the bottom left of your VS code window)
2. Open up the VS code extensions tab, type “Python Environment Manager”. Click Install. This will install the environment manager remotely to your Odhar workspace, rather than

onto your computer like previously

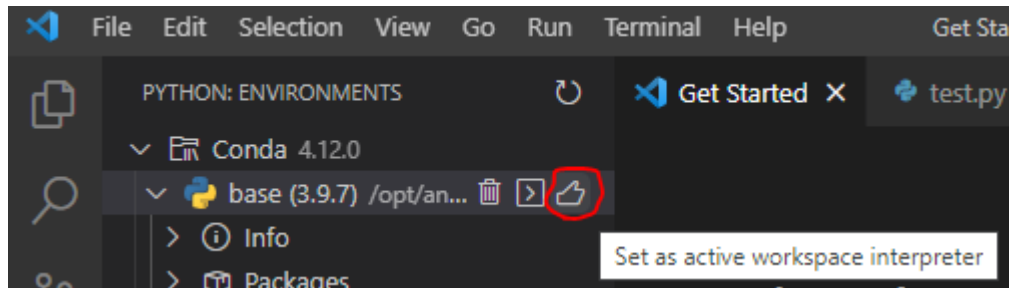


3. This will add a new icon shaped like the python logo on the left sidebar. Here you can find all your available python environments

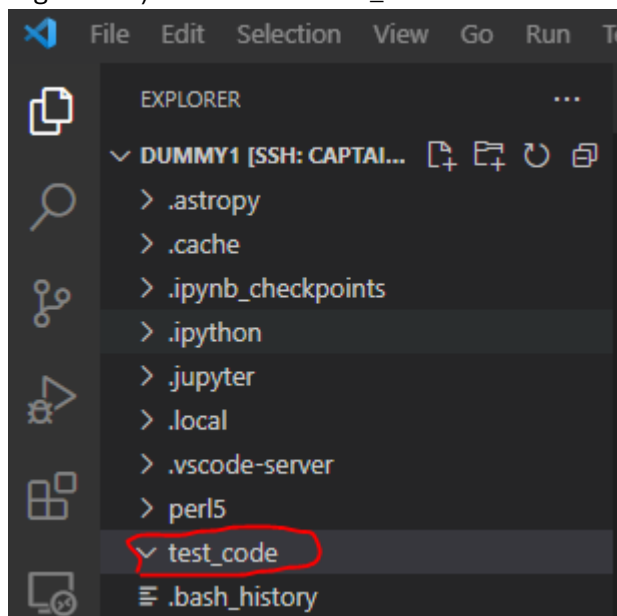


4. As we want a local environment, click “Conda 4.12.0” (or whatever version of Conda it currently says), hover over “base” and click the thumbs up. This will set “base” to be your default python environment, allowing you to install packages you need locally to

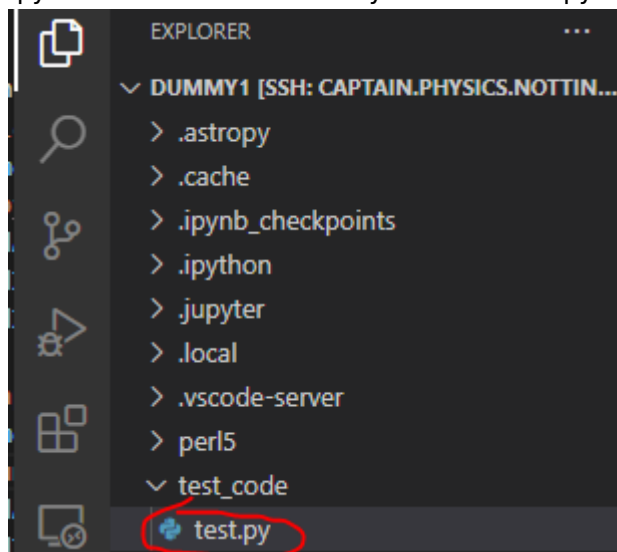
“base”.



5. Restart VS code to make sure the changing of your environment has worked. This is a bit tedious to do every time but as you don't have to change environment very often we will tolerate it.
6. Now VS code has restarted, open your root directory in ODHAR as you have done before. Time to test python code is working. First let us create a new folder (because we are organised!) and name it “test_code”



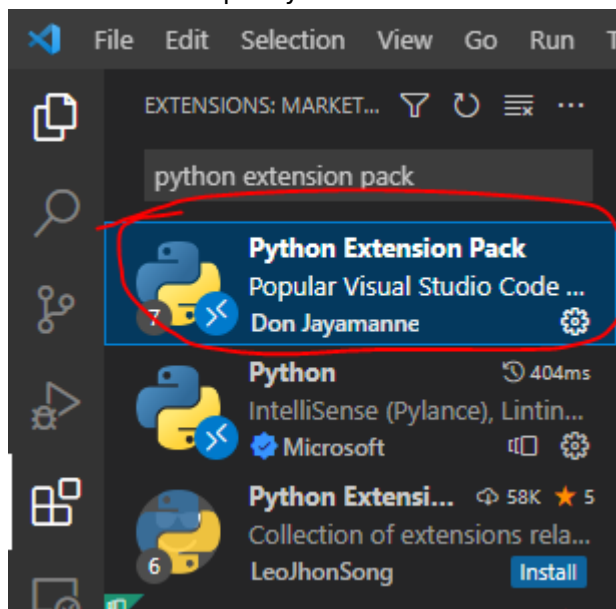
7. Right click your new folder and create a python file named “test.py”. As the extension is .py VS code will automatically know this is a python file



8. In this new python file, add the code:

```
### block 1
import sys
print(sys.version)
### block 2
import numpy as np
test_value=np.arange(10)*np.pi
print("test value is:", test_value)
```

9. You may notice that python isn't actually installed (the run button we've used previously isn't there anymore). While it was installed on your local computer, this is ODHAR.
10. Open up the extensions tab on the left sidebar, search "Python extension pack". Install this extension. The Python extension pack actually installs python and lots of other useful extensions for programming in python, you can click on it to learn a bit more about what it installs. You could install python the same way we did locally, this way just adds some nice quality of life features for us.



11. Now python is installed, you should see your little "run" button has returned in your "test.py" file! Running it using this button gives us an output in the terminal. Here we see the version of python we are running, as well as some multiples of pi.

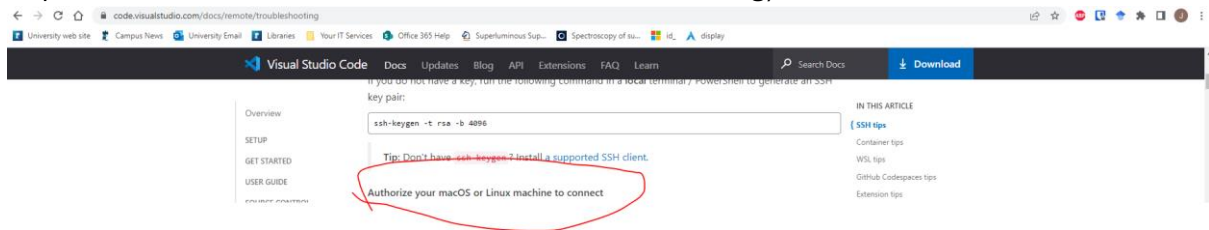
```
(base) dummy1@captain:~$ /opt/anaconda3/bin/python /home/dummy1/test_code/test.py
3.9.7 (default, Sep 16 2021, 13:09:58)
[GCC 7.5.0]
test value is: [ 0.          3.14159265  6.28318531  9.42477796 12.56637061 15.70796327
 18.84955592 21.99114858 25.13274123 28.27433388]
(base) dummy1@captain:~$
```

12. We can now also test Jupyter Notebooks by pressing the "Run Cell" buttons on both parts of our code. As previously, this will open a new window. The output should be the same as what we got in the previous step.

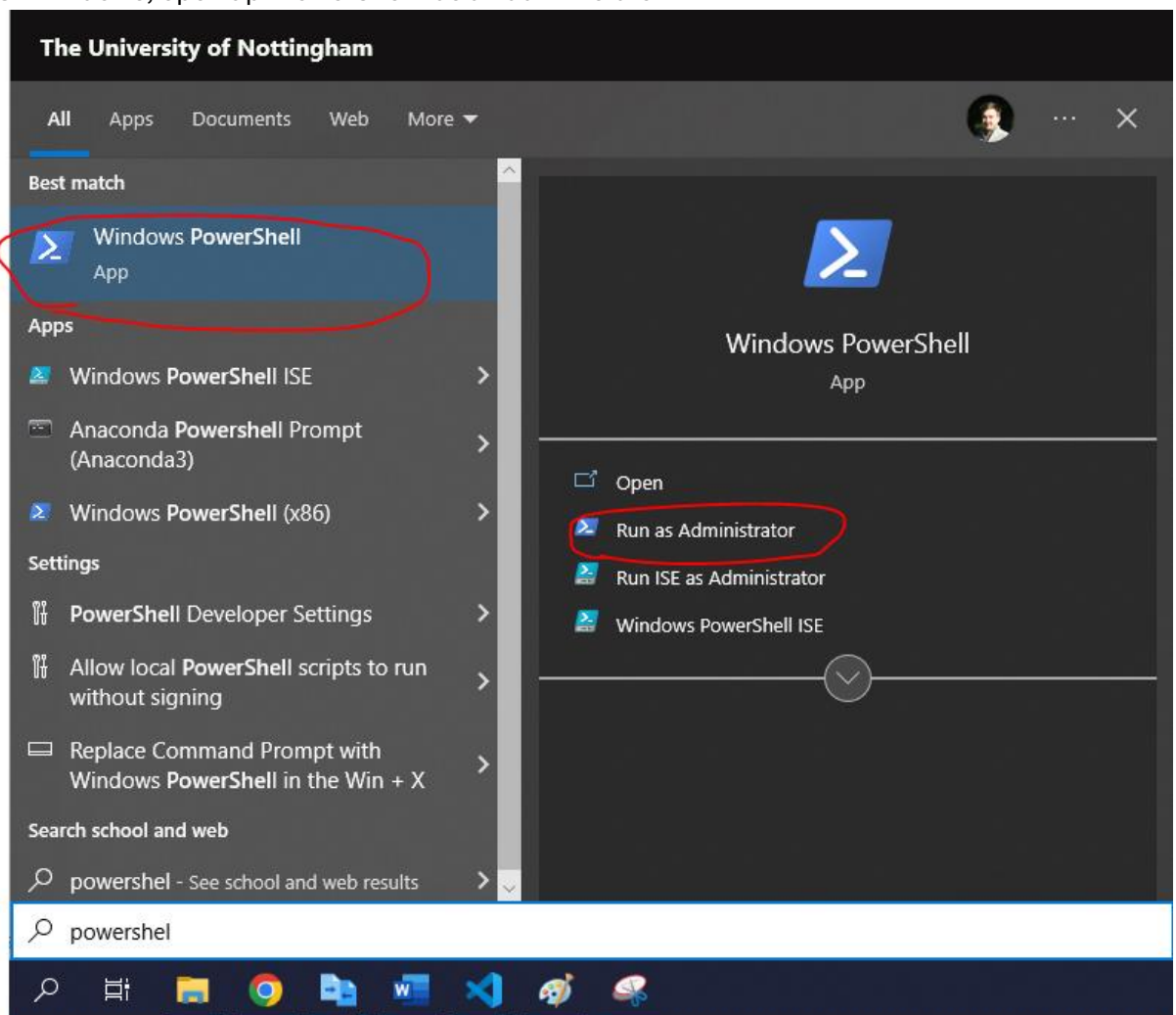
You are now all set up to run python on ODHAR. However, there are two snags: You have to enter your password a bunch when logging in, and you cannot stream graphs you plot remotely to your local machine.

Step 8: Removing the need to enter your password on a trusted machine

If you don't like having to enter your password upon SSHing into Odhar, or if you wish to see any graphs you plot remotely. Follow these steps (on a windows machine, for linux and mac go to <https://code.visualstudio.com/docs/remote/troubleshooting>)



1. On windows, open up "PowerShell" as an administrator



- If you don't have a public key (if you don't know what that is, you probably don't have one!) type the command "ssh-keygen -t rsa -b 4096" into the PowerShell terminal and press enter. It will ask you to enter the file in which to save the key, leave this blank and just press enter. It will ask you to enter a passphrase, leave this blank and press enter. It will ask you to reenter the passphrase, again leave this blank and press enter. This will generate a 4096 long "password" that only your computer and Odhar will know.

```
Administrator: Windows PowerShell

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\WINDOWS\system32> ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (C:\Users\ppyjh10\.ssh\id_rsa):
C:\Users\ppyjh10\.ssh\id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:\Users\ppyjh10\.ssh\id_rsa.
Your public key has been saved in C:\Users\ppyjh10\.ssh\id_rsa.pub.
The key fingerprint is:
SHA256:WJIY32RvIacXow9N42eY8wzy3DuMxcLOPxO/V20h6os ad\ppyjh10@LUIN72326
The key's randomart image is:
+---[RSA 4096]-----+
|
|  .
| o o + o *
| + B . . B B
| o S o * @ +
| . @ #.
| + X O
| .. + =+
| E.o. .+*
+---[SHA256]-----+
PS C:\WINDOWS\system32>
```

- Now enter the line **\$USER_AT_HOST="username@odhar.physics.nottingham.ac.uk"**, where instead of "username" add your own username used to login to Odhar (e.g. ppyjh10 for me). If you were doing this for a different remote machine, replace the Odhar address with that machine's address (e.g. "captain.physics.nottingham.ac.uk"). This creates a PowerShell variable that we can use to make later commands smaller.
- Enter the line **\$PUBKEYPATH="\$HOME\.ssh\id_rsa.pub"**. This is the directory where our key was stored when we generated it.
- Finally, enter the command **\$pubKey=(Get-Content "\$PUBKEYPATH" | Out-String); ssh "\$USER_AT_HOST" "mkdir -p ~/.ssh && chmod 700 ~/.ssh && echo '\${pubKey}' >> ~/.ssh/authorized_keys && chmod 600 ~/.ssh/authorized_keys"** (all one line). This complex command logs into Odhar using your username, and copies the keyfile you made over to the correct place
- If everything has worked correctly, it should ask you to enter if you trust odhar (enter "yes") and for your password. Once these are entered you have successfully transferred your key file over to the right place

```
PS C:\WINDOWS\system32> $USER_AT_HOST="dummy1@captain.nottingham.ac.uk"
PS C:\WINDOWS\system32> $PUBKEYPATH="$HOME\.ssh\id_rsa.pub"
PS C:\WINDOWS\system32> $pubKey=(Get-Content "$PUBKEYPATH" | Out-String); ssh "$USER_AT_HOST" "mkdir -p ~/.ssh && chmod 700 ~/.ssh && echo '${pubKey}' >> ~/.ssh/authorized_keys && chmod 600 ~/.ssh/authorized_keys"
The authenticity of host 'captain.nottingham.ac.uk (128.243.43.141)' can't be established.
ECDSA key fingerprint is SHA256:5MBKMEy9MHZEqzuxO/efictIxx7zb/+u3ZQoEMKoY.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'captain.nottingham.ac.uk' (ECDSA) to the list of known hosts.
dummy1@captain.nottingham.ac.uk's password:
Permission denied, please try again.
dummy1@captain.nottingham.ac.uk's password:
PS C:\WINDOWS\system32>
```

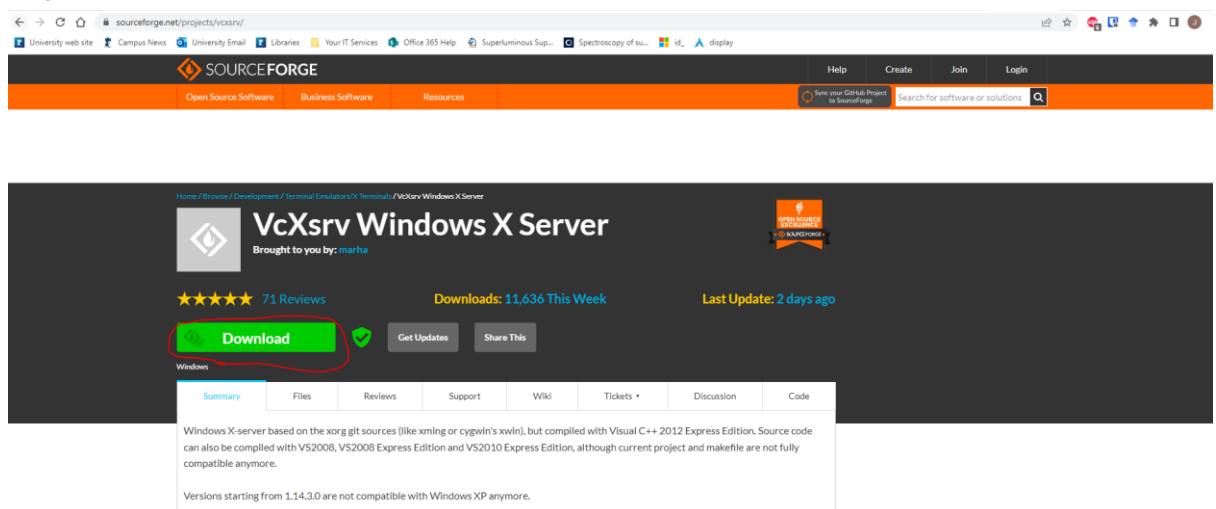
7. Try SSHing into ODHAR using VS code, if everything works you wont have to enter your password!

If these commands seem a bit spooky, just know the ssh-keygen generates a random collection of 4096 numbers and letters, the next 4 commands just send that string safely to Odhar, and when logging into ODHAR the computers compare these two random strings to see if they match, rather than comparing a password.

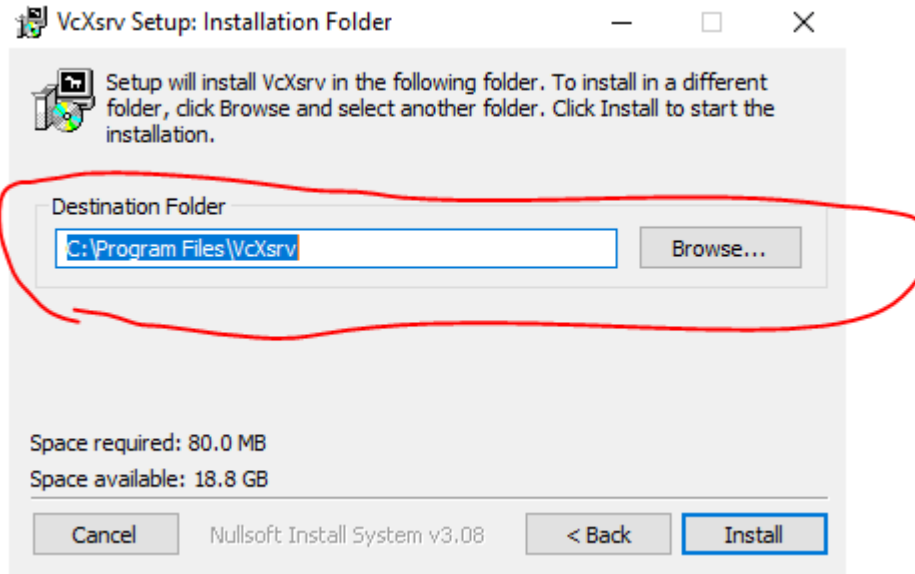
Step 9: Setting up X11 Forwarding

This is the final step to getting our beautiful plots out of Odhar. If you are using Jupyter notebooks (or the code blocks) you can ignore this step, only if you are running code files directly do you need to follow this step. You **must** have done the previous step (removing the need for passwords) before doing this step as the X11 forwarding extension we use relies on key file to transfer data.

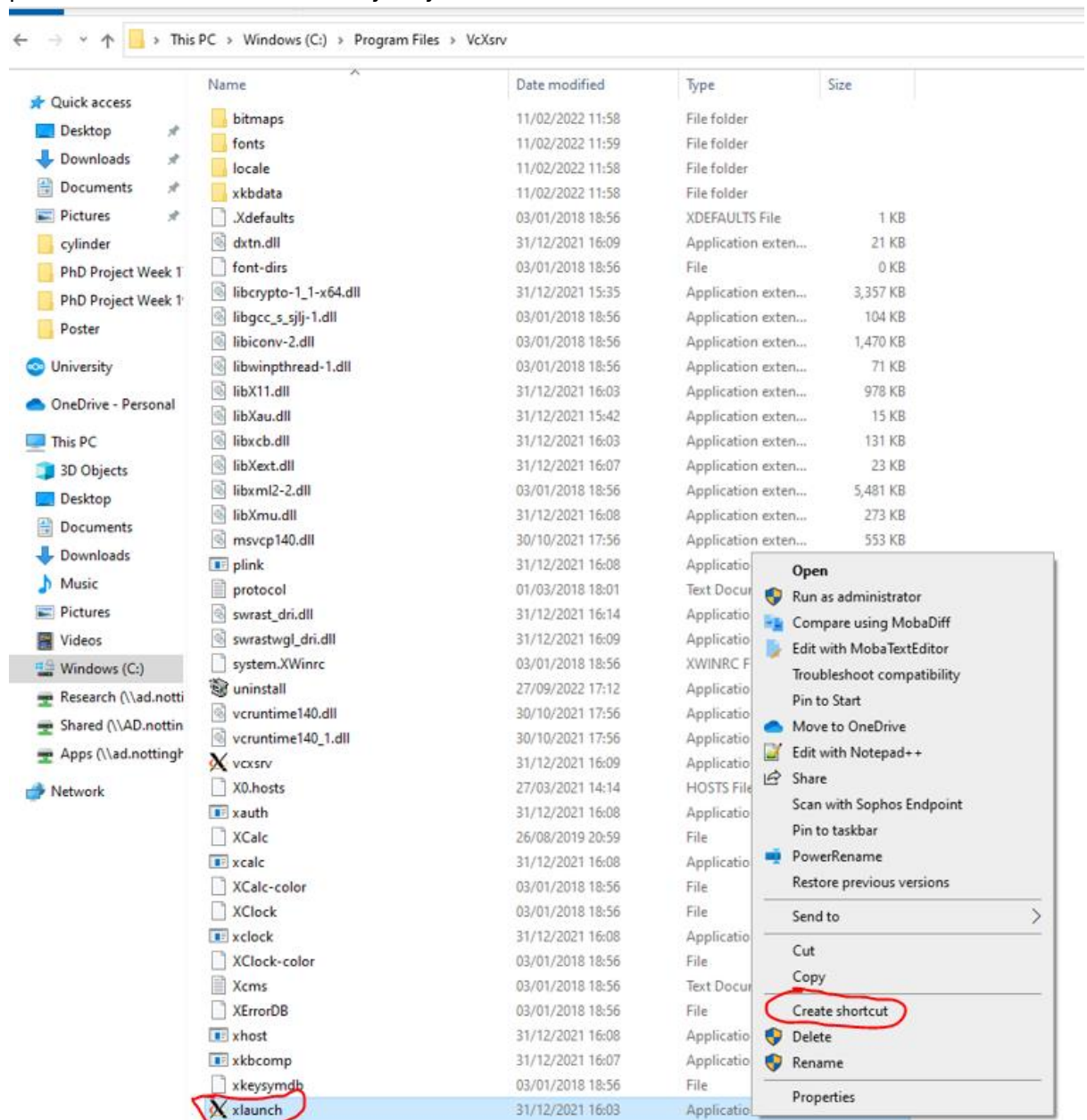
1. If you are on windows, navigate to <https://sourceforge.net/projects/vcxsrv/> and download VCXSRV. This is windows native X11 server. You need one of these as X11 is a unix based software. If you are on Linux (or possibly mac) you wont need software like this.



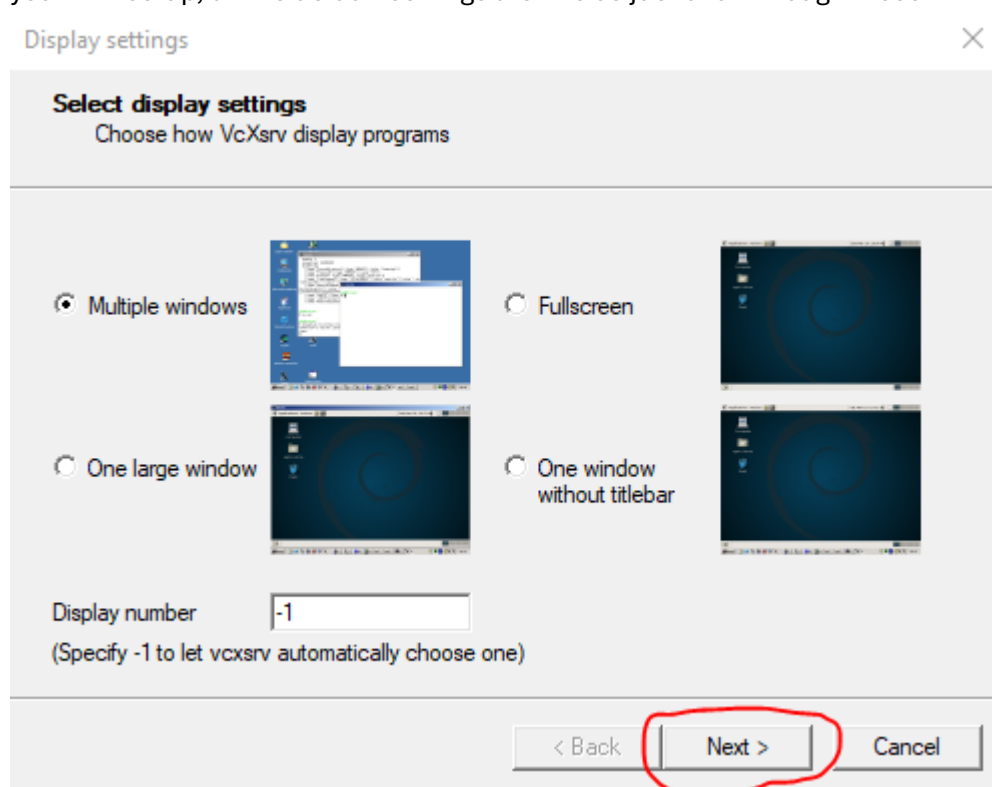
2. Run the VCXSRV installer, take note of where it installs



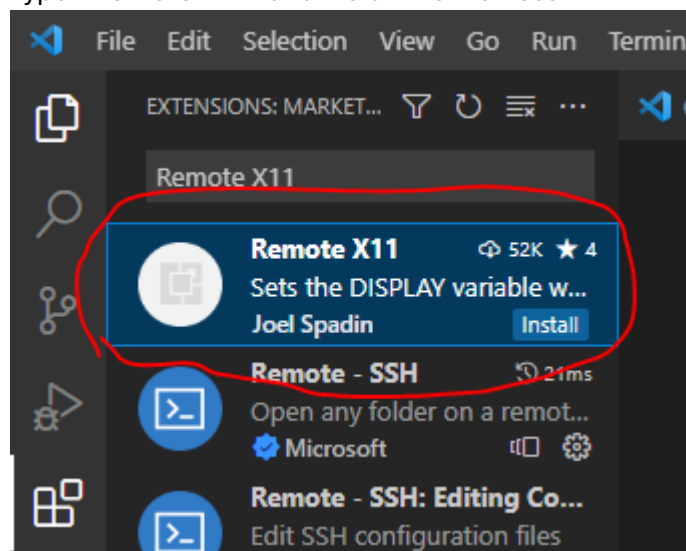
3. Once installed, navigate to the install location create a shortcut of “xlaunch.exe” and put the shortcut somewhere easy for you to find



- Click the shortcut you created, this will open a dialog asking you how you want to setup your X11 setup, all the default settings are fine so just click through these



- You will need to run this server each time you start up your computer.
- Open VS code and SSH into Odhar
- Open the extensions tab from the left sidebar
- Type **"Remote X11"** and install the first result



- Once installed, fully restart VS code and reconnect to Odhar
- You can test everything is working by opening the terminal (**Ctrl + Shift + `**) and entering the command keys
- This should bring up a pair of eyes that watch your cursor ominously...

If you see the ever-watching eyes, you have successfully installed your X11 server on both ends, if you run commands like `plt.show()` they will be sent to your local machine to be rendered (for very large plots this can take some time, but you should get an empty window while it loads).

If you don't see any eyes or plots, but there aren't any errors in console, you need to restart your X11 server by clicking the shortcut we created

Conclusions

You should now have successfully installed and setup VS code. While there were quite a few steps, setting up your work environment like this vastly speeds up your workflow as a PhD. When you spend less time spent banging your head against the wall trying to get code the run and download the results, you have plenty more time to bang your head against the wall trying to understand what your code actually outputted.

If you have any questions, or if part of this guide is outdated, please don't hesitate to email me at: ppyjh10@nottingham.ac.uk. Or just speak to me in person...