Remotely work with Jupyter (IPython) Notebooks at UCL Geography

Chad Stainbank

March 21, 2017

1 Introduction

The Jupyter Notebook (previously IPython Notebook) provides a feature-rich interactive environment for learning and using Python. If you're studying for a BSc or MSc at the University College London Department of Geography then it's likely that you will have worked with these Notebooks in one or more of your modules or in the preparation of a dissertation, usually while seated at one of the workstations in Pearson 110a that form the Geography Linux Cluster. Since physical access to Pearson 110a is not always practical, you may wish to have *remote* access to these files from another computer. This guide will show you how to work with Notebooks, hosted on the Linux Cluster, from your own computer.

The bulk of this guide is written for somebody using the terminal on a Linux computer. However, it is likely that your own computer runs on either macOS or Microsoft Windows. Since macOS is a Unix-based operating system, Apple fans should be able to enter all the commands in Section 2 into the Mac Terminal with little modification. Windows users, on the other hand, must first set up a terminal emulator before following these instructions, as explained in Section 3.

2 On Linux/ Mac OS X

Section 2.1 provides a very explicit set of instructions to get up and running with remote Jupyter Notebooks using the Linux terminal. If you're familiar with the command-line interface then you could skip to Section 2.2, where these commands are condensed to shell functions, although it's worth going through this once to ensure that everything is working smoothly.

2.1 Manual

The *Listing* boxes, such Listing 1 below, display lines of commands to be entered into the terminal. If a command contains text enclosed in angle brackets (<>) then you must replace the variable (including the brackets) as specified. Any text following the # character is a comment and does not need to be entered. Many commands take *flags* to specify some of their behaviour — be careful to distinguish between the single dash (-) and the double dash (--) when typing these out.

Listing 1: An example of a command

```
cmnd --flag <variable> # this does something
```

2.1.1 Log in to cluster machine

The program SSH allows you to log into one of the machines in Pearson 110a from your own computer. The Linux cluster is set up in a way that requires a two-step (or *multi-hop*) remote login, meaning that you must first pass through a *gateway server*. To make the first hop, open a terminal window and enter the command in Listing 2, replacing `<USER>` with your UCL username and `<GATEWAY>` with the name of any one of the Geography gateway servers listed on the Geography Remote Access page).

Listing 2: Login to gateway

```
ssh <USER>@<GATEWAY>.geog.ucl.ac.uk
```

The first time you log in to a given machine, you will get the following message:

```
The authenticity of host '<GATEWAY> (...)'  can't
be established.

RSA key fingerprint is ...

Are you sure you want to continue connecting (yes/no)?
```

Type `yes` to confirm, then enter your Geography cluster password [1] when prompted to log in to the gateway server.

To make the second hop, enter the command in Listing 3, replacing `<MACHINE>` with the name of your favourite teaching workstation in Pearson 110a. Notice that, as you are already within the geography network, you don't have to specify `<USER>` or finish the address with `.geog.ucl.ac.uk` this time.

Listing 3: Login to workstation

```
ssh <MACHINE>
```

Simply enter your password again to complete the login to your chosen cluster machine.

2.1.2 Run remote Notebook Server

Start running a remote Notebook server with the command in Listing 4. The flag `--no-browser` prevents Firefox from opening automatically, while `--port` tells the server to listen on a particular port number. This is set to 8888 by default, although you can choose just about any integer above 1026 to replace `<PORT>`.

Listing 4: Run Notebook server

```
jupyter notebook --no-browser --port=<PORT>
```

Jupyter should now inform you that:

[1] This is not necessarily the same password that you use for other UCL services.

```
The IPython Notebook is running at:  http://localhost:<PORT>.
```

Look carefully at the link and check that the port number is the same the one you specified. If it has changed, simply note down the new number and use it as `<PORT>` for all subsequent commands.

2.1.3 Set up SSH tunnel

The next step is to open an *SSH tunnel* with the workstation in Section 2.1.1, allowing you to connect to the Notebook server created in Section 2.1.2. This is achieved by passing the `-L` flag, which tells SSH to use local port forwarding to transfer data between nodes. In a *new* terminal, enter the commands in Listings 5 and 6, replacing variables and supplying passwords as before.

Listing 5: Tunnel to gateway server

```
ssh -L <PORT>:localhost:<PORT> <USER>@<GATEWAY>.geog.ucl.ac.uk
```

Listing 6: Tunnel to cluster machine

```
ssh -L <PORT>:localhost:<PORT> -N <MACHINE> # this will hang
```

If all has gone well then the terminal will appear to be hanging, as the optional `-N` flag prevents any more user input. This means that the SSH tunnel is now open for business.

2.1.4 Open Notebook in browser

To use the Notebook, simply open the link from the first terminal (Section 2.1.2) using a browser on your local computer. Congratulations: you should now be looking at the familiar Jupyter Notebook file browser. If something has gone wrong, see Section 4, otherwise move on to Section to 2.2 to make the whole remote Notebook process easier.

2.2 Automatic

2.2.1 Generate SSH keys

Following all the steps in 2.1 requires you to enter your password a total of four times (once for each SSH command). *SSH keys* provide an alternative form of authentication, using a private and a public key to allow passwordless login over SSH. Two seperate pairs of keys must be set-up: one for the initial hop from your computer to the gateway node and another for the onward hop to the workstation.

To set up the first pair of keys, open a terminal on your computer and enter the command:

Listing 7: Generate SSH keys

```
ssh-keygen -t rsa
```

Let the program use the default file with no passphrase by presssing *Enter* three times, then copy the public key to the geography file network using Listing 8 and following the instructions returned by the program.

Listing 8: Copy public key to geography file system

```
ssh-copy-id <USER>@<GATEWAY>.geog.ucl.ac.uk
```

To generate a pair of keys for the second hop, simply login to the any Geography cluster machine and reenter the commands in Listings 7 and 8.

2.3 Geography login/tunnel functions

It will very quickly become tedious to perfectly type out the commands in Section 2.1 every time you want to run a Notebook remotely. The two functions in Listing 12 (at the end of this document), written for the *bash shell*, condense most of this to a few short commands.

Be aware that Listing 12 is for demonstration only; it does not display properly in a pdf document and so will not work if used as-is. Instead, a raw text file containing these functions can be found at:
`https://github.com/stainbank/remote_ipynb/blob/master/geog_functions.sh`

The simplest way to "install" these two functions is to paste the entire code block — replacing the defaults as appropriate — at the end of a file in your home directory called .bashrc file. If this file doesn't already exist, as is the case on Mac OS X, simply create it and then add the following line to the file .bash_profile (also in the user root directory):

Listing 9: Source .bashrc on startup

```
if [ -f ~/.bashrc ]; then . ~/.bashrc; fi
```

The command *geog* replaces the commands in Section 2.1.1, while *geogport* replaces Section 2.1.3. You can therefore work with a remote Notebook by simply entering:

Listing 10: Set up and tunnel to remote Notebook server

```
# In one terminal
geog
jupyter notebook --no-browser --port=<PORT>
# In another terminal
geogport
```

To allow you to change any variables from their default values, these functions also take positional arguments, in the order:

`geog <MACHINE> <GATEWAY> <USER>`

`geogport <PORT> <MACHINE> <GATEWAY> <USER>`

Note that you must supply all *preceeding* arguments, and prefix `$` to the name of an argument to reuse its default value. Some usage examples are provided in Listing 11

Listing 11: Examples of arguments to geography login/tunnel functions

```
geog ulanbator # change machine
geog $MACHINE triangle ucfaxyz # change user and gateway
geogport 8889 # change port
geogport $GPORT lima # change machine
```

3 On Windows

Sections 3.2 and 3.3 describe how to setup a Windows terminal emulator, into which you can enter commands adapted from Section 2. Since much of the material here refers to that section, you must read it first. However, I strongly urge any Windows users intending on working extensively with Python and Jupyter Notebooks to consider installing a Linux distribution...

3.1 Install Linux

Don't be alarmed: this need not be a drastic switch as there are two simple ways to set up Linux *alongside* Windows. The first is to use a *virtual machine*, an OS that runs *inside* your current system. This is very simple to setup (see here) and I myself use this method to work from UCL's library computers.

The second option, and the one I use for my personal computer, is *dual-booting*. Although this imay seem somewhat involved, a native Linux install is always faster than a virtual one and there are plenty of guides available online to guide you through the process (e.g. 1, 2, 3, 4).

Whichever installation option you go for, you will need to choose one of the many *distributions* of Linux. Ubuntu and Linux Mint both appeal to new Linux users, and any support pages for one usually applies to the other. For use on a virtual machine or an ageing laptop, I would recommend opting for a more lightweight flavour — i.e. one that features MATE, Xfce or LXDE as it's desktop environment.

3.2 Windows Subsystem for Linux

If you have Windows 10 then you'll be able to take advantage of a new Microsoft development called *Windows Subsystem for Linux* (WSL), otherwise referred to as *Bash on Ubuntu on Linux*, which provides a Linux command-line interface on Windows. To use WSL to run Jupyter Notebooks, simply install the feature and use these Bash terminals to follow the instructions in Section 2

3.3 PuTTY

If you don't have access to WSL then you'll have to use PuTTY, a popular terminal emulator and SSH client, to access your Notebooks. First install the program from its website, then run a PuTTY instance to load its GUI configuration screen. Navigate to the *Session* screen and enter `<USER>@<GATEWAY>.geog.ucl.ac.uk` in the *Host Name* box, making the appropriate replacements (described in Section 2). [2] To store these gateway login settings, enter a name under *Saved Sessions* and click *Save* (Figure 1).

To set up the SSH tunnel, ensure that the settings from the previous session are loaded, then navigate to the port forwarding options via *Connection > SSH > Tunnels* (Figure 2). Under *Source Port* supply `<PORT>`, and as the *Destination* type `127.0.0.1:<PORT>`, substituting <PORT> with a valid port number as described in Section 2.1.2. Click the *Add* button to commit this information to the list of *Forwarded ports*. Finally, to save these settings,

[2]Follow this guide if you want to be able to load X11 windows (i.e. graphical programs such as *gedit* or *Firefox*), although this is not necessary for using Notebooks.

navigate back to *Sessions* and enter a *new* name under *Saved Sessions* before clicking *Save* (Figure 3).

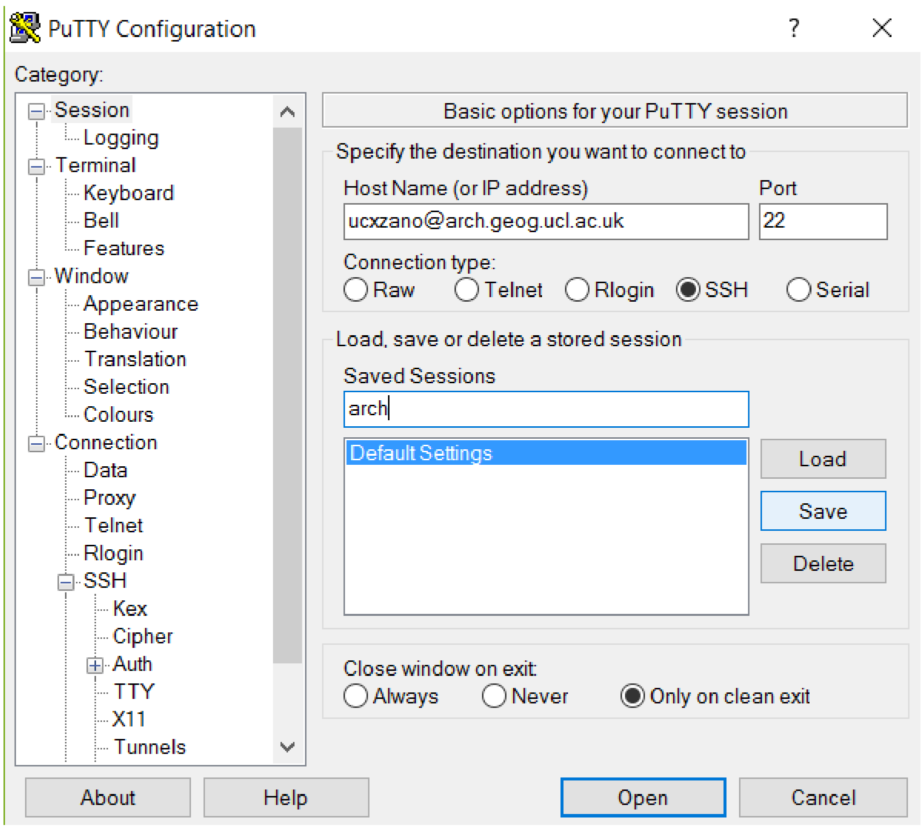
To work with Jupyter Notebook on the Geography Linux Cluster, open each of these session profiles in individual PuTTY windows. The first session you created (*arch* in Figure 1) replaces the command in Listing 2, while the second (*archport* in 3) replaces Listing 5. Therefore, to remotely work with a Jupyter Notebook, simply substitute these while following the instructions in Section 2.1.

4 Troubleshooting

Occasionally, when trying to set up the SSH tunnel, you will get a message like:

```
bind: Address already in use
channelsetupfwd_listener: cannot listen to port: <PORT>
Could not request local forwarding
```

This means that the tunnel could not be established as another process is using that particular port. If this occurs, start everything over again with a different port number.



Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): ucxzano@arch.geog.ucl.ac.uk

Port: 22

Connection type: Raw Telnet Rlogin [✓ SSH] Serial

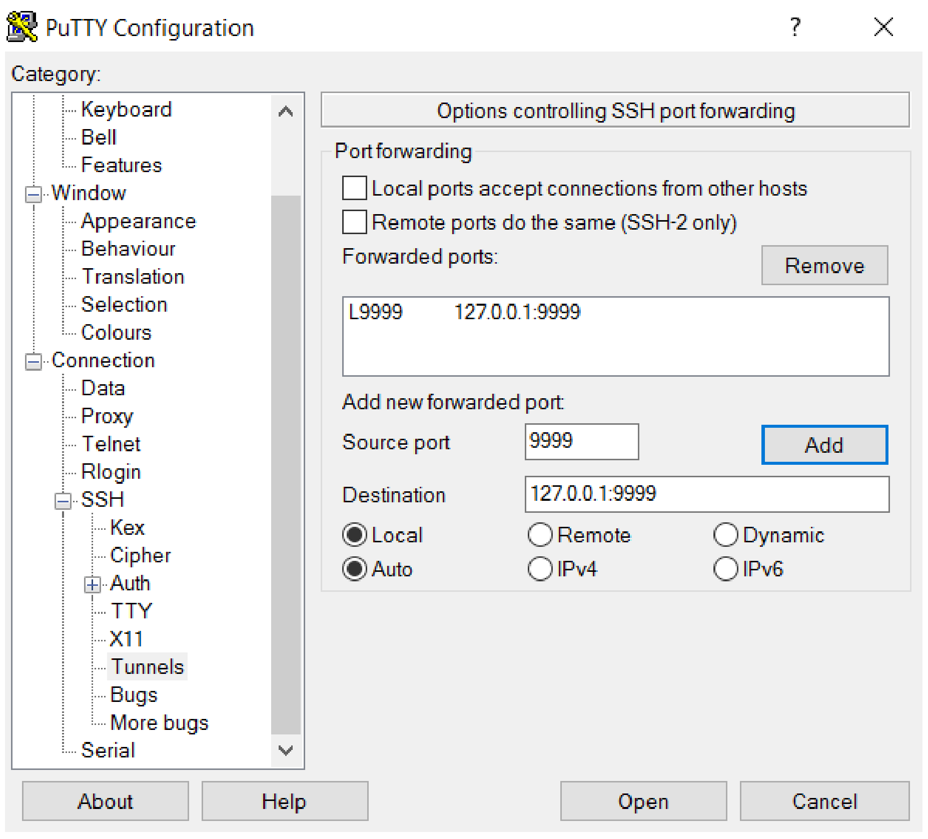
Load, save or delete a stored session

Saved Sessions: arch

Default Settings

Close window on exit: Always Never [✓ Only on clean exit]

Figure 1: Save session to login to gateway



Options controlling SSH port forwarding

Port forwarding

☐ Local ports accept connections from other hosts

☐ Remote ports do the same (SSH-2 only)

Forwarded ports:

```
L9999       127.0.0.1:9999
```

Add new forwarded port:

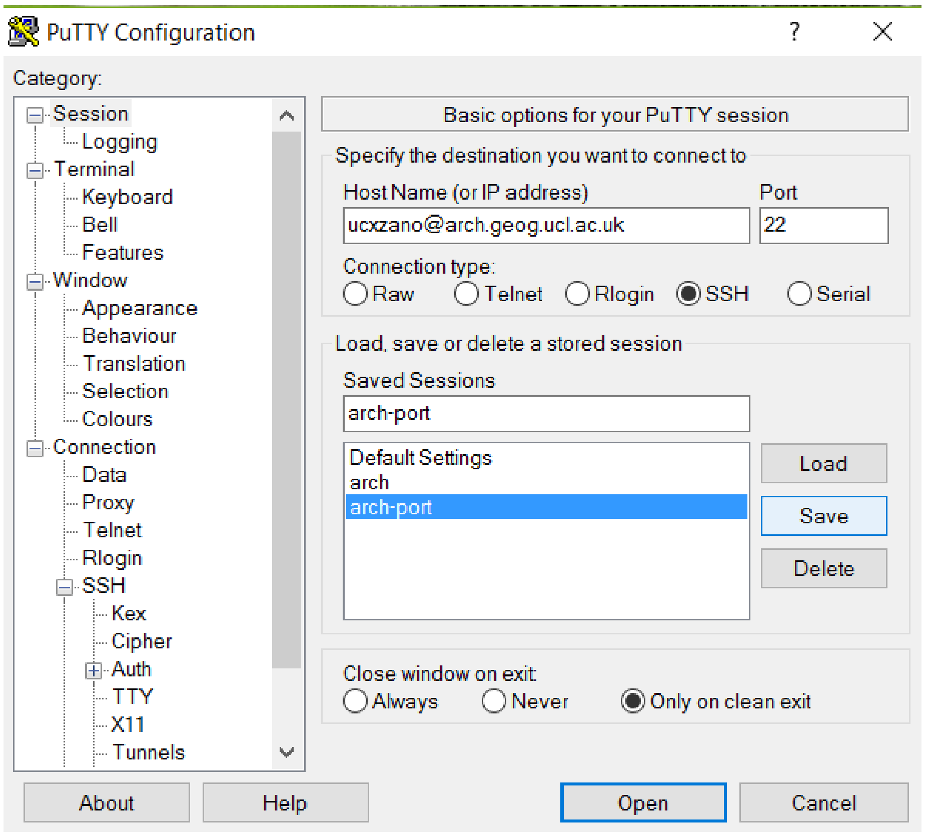
Source port: 9999

Destination: 127.0.0.1:9999

[● Local] ○ Remote ○ Dynamic

[● Auto] ○ IPv4 ○ IPv6

Figure 2: Set up port forwarding to gateway



Specify the destination you want to connect to

Host Name (or IP address): ucxzano@arch.geog.ucl.ac.uk

Port: 22

Connection type: Raw Telnet Rlogin [✓ SSH] Serial

Load, save or delete a stored session

Saved Sessions: arch-port

Default Settings
arch
arch-port

Close window on exit: Always Never [✓ Only on clean exit]

Figure 3: Save session to tunnel to gateway

Listing 12: Functions to login/tunnel to geography cluster

```
# Replace these defaults
GUSER=ucaaxyz
GATEWAY=shape
MACHINE=city
GPORT=number
# Log in UCL Geography Linux cluster machine
geog () {
    ssh -t -Y ${3:-$GUSER}@${2:-$GATEWAY}.geog.ucl.ac.uk ssh -Y
         ${1:-$MACHINE}
}
# Tunnel into UCL Geography Linux cluster machine
geogport () {
    PORT=${1:-$GPORT}
    ssh -L $PORT:localhost:$PORT ${4:-$GUSER}@${3:-$GATEWAY}.
        geog.ucl.ac.uk ssh -L $PORT:localhost:$PORT -N ${2:-
        $MACHINE}
}

# Tunnel into UCL Geography Linux cluster machine
geogport () {
    PORT=${1:-$GPORT}
    ssh -L $PORT:localhost:$PORT ${4:-$GUSER}@${3:-$GATEWAY}.
        geog.ucl.ac.uk ssh -L $PORT:localhost:$PORT -N ${2:-
        $MACHINE}
}
```