Remotely work with Jupyter (IPython) Notebooks at UCL Geography

Chad Stainbank

March 16, 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've encountered these Notebooks in one or more of your modules or in the preparation of a dissertation. Usually you will work with them 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 Mac OS X (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

2.1 Manual

This section provides a very explicit set of instructions to get up and running with remote Jupyter Notebooks. The Listings boxes display commands to be entered into the terminal. <> enclose values which must be replaced as specified, while comments — any text after # — do not need to be entered. These commands may not render properly in pdf so be careful to distinguish between the single dash (-) and the double dash (--). 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.

2.1.1 Run Notebook Server

The first step is use the program SSH to remotely log into one of the machines in Pearson 110a and start up a Notebook server. Because of the way the Linux cluster is set up, this requires a two-step (or *multi-hop*) login. To make the first hop, open a new terminal and enter the command in Listing 1, replacing <USER> with your UCL username and <GATEWAY> with the name of any one of the gateway servers listed on Geography Remote Access page.

Listing 1: Login to gateway

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

Enter your Geography network password [1] when prompted, followed by the command in 2. Here, <MACHINE> must be replaced by the name of your favourite teaching workstation in Pearson 110a.

Listing 2: Login to workstation

```
ssh <MACHINE>
```

After entering your password again, start running the Notebook server with command 3. The flag "–no-browser" prevents Firefox from opening automatically, while "–port" tells the server to listen on a particular port number. This is 8888 by default, although you can choose just about any integer above 1026 to replace <PORT>.

Listing 3: Run Notebook server

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

Jupyter should now inform you that "The IPython Notebook is running at: <link>". Without opening the link, check that the port number (after "http://localhost:") 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.

2.1.2 Set up SSH tunnel

The next step is to open an SSH tunnel, through which you can connect to the freshly created Notebook server. In a **new** terminal, enter the commands in Listings 4 and 5. Here, the "-L" flags tell SSH to use local port forwarding to transfer data between nodes. As before, replace the bracketed values and supply your password when prompted.

Listing 4: Tunnel to gateway server

```
ssh -L <PORT>:localhost:<PORT> -N <MACHINE>
```

Listing 5: Tunnel to machine

```
ssh -L <PORT>:localhost:<PORT> -N <MACHINE>
```

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

If all has gone well, the terminal will appear to be hanging [2] and the SSH tunnel should be open. To use the Notebook, copy the link produced in Section 2.1.1 (e.g. `http://localhost:8888`) and paste it into the URL bar of your favourite browser. Congratulations, you should now be looking at the familiar Jupyter Notebook file browser. If not, see 2.1.3

2.1.3 Troubleshooting

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

> bind: Address already in use
>
> channel*setup*fwd_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.

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 6: Generate SSH keys

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

Let the program use the default file with no passphrase by presssing enter twice, then copy the public key to the geography file network using command 7 and following the instructions returned by the program.

Listing 7: 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 machine on the linux cluster and repeat the process. [3]

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 8, written for *bash*, condense most of this to a few short commands.

[2] The optional "-N" flag prevents any more user input

[3] You do not need to specify `<USER>`, nor finish the address with ".geog.ucl.ac.uk", as you are already within the geography network.

Listing 8: Functions to login/tunnel to geography cluster

```
# Replace these defaults
GUSER=ucfacms
GATEWAY=arch
MACHINE=freetown
GPORT=9000
# 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}
}
```

Be aware that Listing 8 is for demonstration only; it does not display properly in a pdf document and so will not work if used as-is. Instead, 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 — into 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 Listings 1 and 2, while *geogport* replaces 4 and 5. 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
```

These functions also take positional arguments, in the order:
geog <MACHINE> <GATEWAY> <USER>
geogport <PORT> <MACHINE> <GATEWAY> <USER>
This allows you to change any variables from their default values [4]:

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*,

[4] You must supply all *preceeding* arguments; prefix the an argument name with **$** to use the default value

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). [5] 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* > *Auth* > *Tunnels* (Figure 2). Under *Source Port* supply <PORT> and enter 127.0.0.1:<PORT> as the *Destination*, substituting <PORT> with a valid port number as described in Section 2.1.1. Click the *Add* button to commit this information to the list of *Forwarded ports*. Finally, to save these settings, 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 1, while the second (*archport* in 3) replaces Listing 4. Therefore, to remotely work with a Jupyter Notebook, simply substitute these while following the instructions in Section 2.1.

[5] 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.

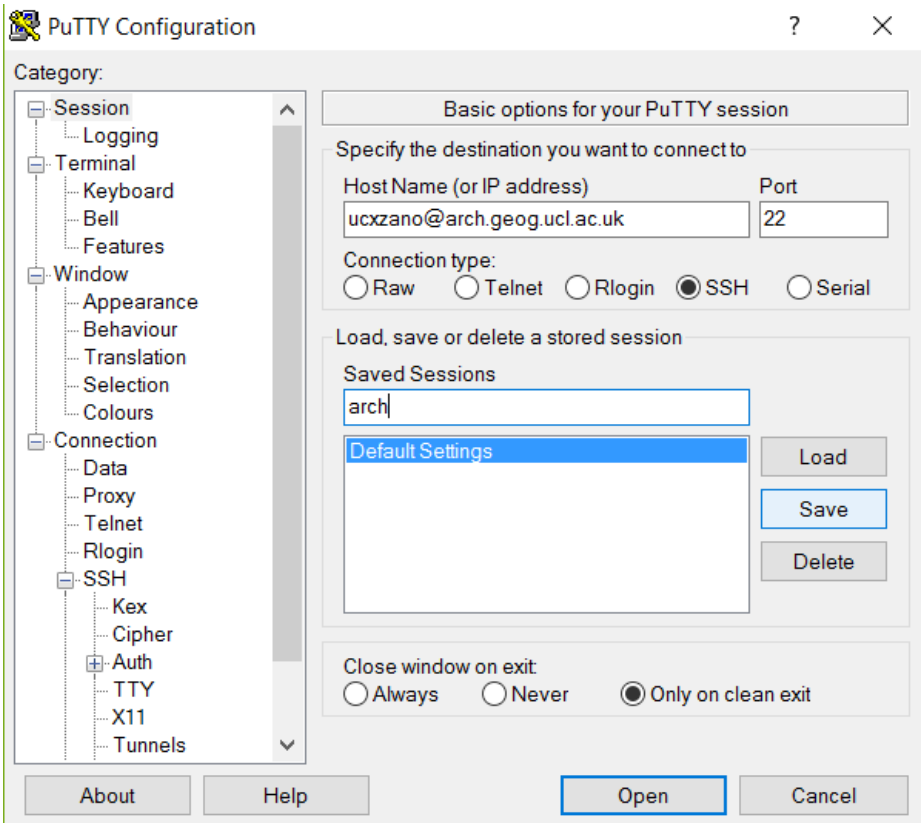

Figure 1: Save session to login to gateway

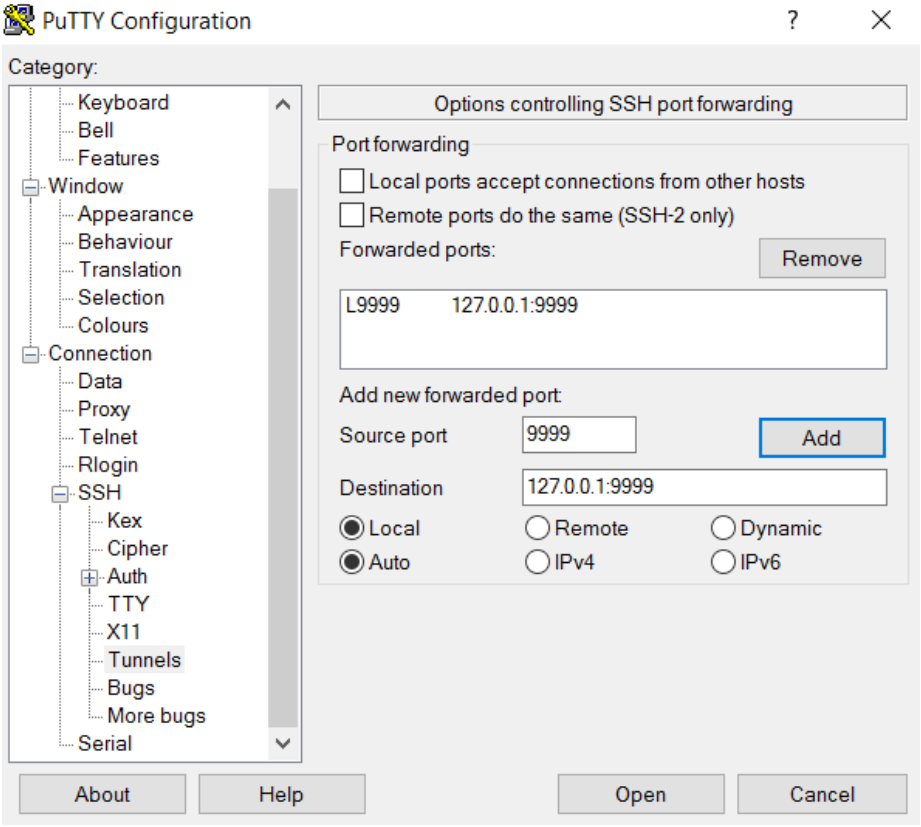

Figure 2: Set up port forwarding to gateway

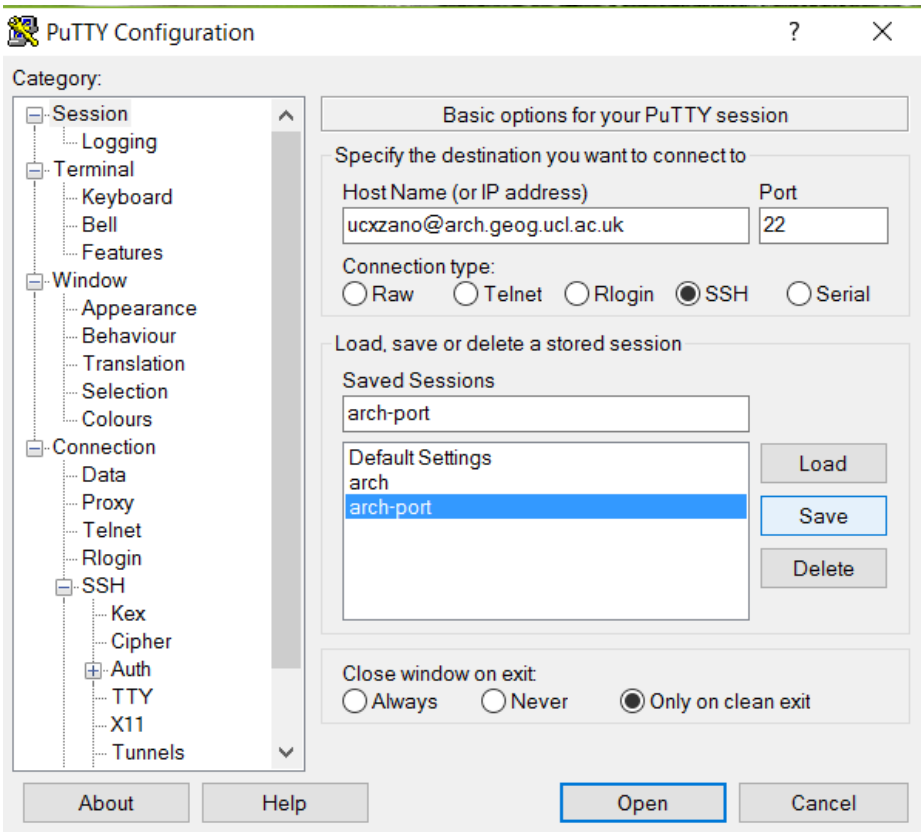

Figure 3: Save session to tunnel to gateway