**SETUP PCR-GLOBWB**

**Exercises (cooking recipes)**

1. **Setup environments for working (e.g. login to velocity and just load existing software)**

1. **Screen (for multi-tasking within your linux terminal)**

1. **Setup environments for working (e.g. login to velocity and just load existing software)**

These exercises assume that we use “velocity” as our computing machine. Please follow the guidance below to create a SSH connection to “velocity” from your laptop/computer.

PS: The following examples use **“mobaxterm”**, which is recommended if your laptop/computer runs on Windows. You can install “mobaxterm” from <https://mobaxterm.mobatek.net/download.html>. You can just use the **“Free”** edition, but please use/install the **“Installer edition”** (the “Portable edition” is not recommended).

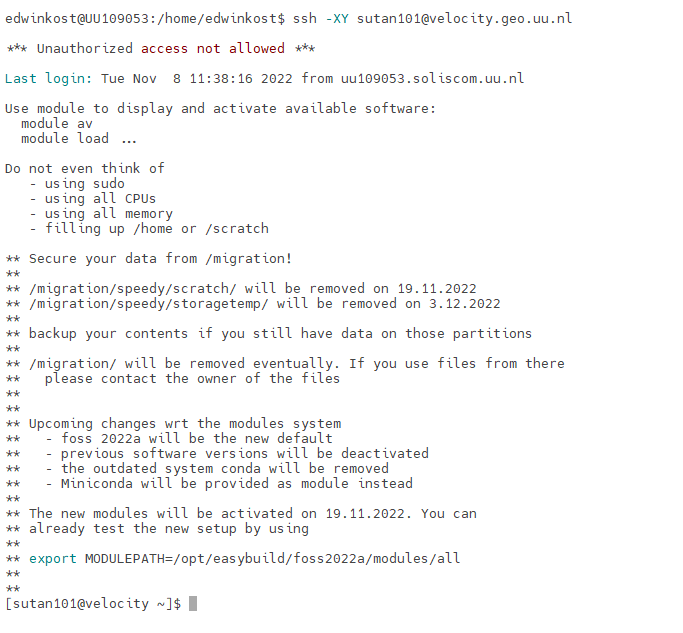
1. **Create a SSH connection**. Open your mobaxterm (on your laptop/computer) to create a SSH connection to “velocity” from your mobaxterm terminal.

# on your mobaxterm terminal (on your laptop), make a SSH connection to velocity, please add the option/argument “-XY”, which is needed for visualizing/displaying GUIs/windows from applications that are running on your velocity terminals.

# ssh -XY <your\_velocity\_username>@velocity.geo.uu.nl

$ ssh -XY sutan101@velocity.geo.uu.nl

If the connection is successful, you should arrive on the velocity terminal. Please see the screenshot below for an illustration (it is not going to be exactly the same; but the machine name “velocity” and your user name should appear).



**Leaving (exit) velocity (returning to your mobaxterm terminal)**

You can leave your VM (and return to your mobaxterm terminal), by typing “exit”. See the following for an illustration.

# I’m now still inside my velocity and I want to print my host/machine name

[sutan101@velocity ~]$ echo $HOSTNAME

velocity.geo.uu.nl

# leaving velocity (and returning to mobaxterm terminal)

[sutan101@velocity ~]$ exit

logout

Connection to velocity.geo.uu.nl closed.

# print my host/machine name (from my local mobaxterm terminal)

edwinkost@UU109053:/home/edwinkost$ echo $HOSTNAME

UU109053

1. **Load all software for PCR-GLOBWB**. Next, we will load all software that is required for running and working with PCR-GLOBWB on velocity. For this, you can just use the existing bash script that is stored in the file “/home/sutan101/load\_all\_default.sh”. This will load a conda/python environment that has all modules required for PCR-GLOBWB, such as pcraster, netcdf4, ncview and cdo.

PS: You can also setup (and load) your own PCR-GLOBWB conda environment (e.g. for your own laptop), by following the steps 1-3 of <https://github.com/UU-Hydro/PCR-GLOBWB_model#how-to-install>.

# on velocity, make sure that you are now on your home directory

$ cd $HOME

# print your current working directory

$ pwd

/home/sutan101

# copy /home/sutan101/load\_all\_default.sh to your home directory

$ cp /home/sutan101/load\_all\_default.sh $HOME

# list the content of my home (to check whether the copy is successful)

$ pwd

/home/sutan101

$ ls -l

total 20

drwxr-xr-x 19 sutan101 users 4096 Dec 10 2021 data

drwxr-xr-x 5 sutan101 users 4096 Oct 12 11:04 github

**-rw-r--r-- 1 sutan101 users 583 Nov 8 12:00** **load\_all\_default.sh**

drwxr-xr-x 2 sutan101 users 4096 Sep 6 16:31 tmp

# then, load the file “load\_all\_default.sh” to load the PCR-GLOBWB conda env

$ pwd

/home/sutan101

$ . load\_all\_default.sh

# you can use “geany” (text editor) to check the content of “load\_all\_default.sh”

$ geany load\_all\_default.sh

To this point, we have load all software required. To test them, please type the following commands.

# pcraster, aguila, gdal

$ pcrcalc

$ aguila

$ gdalinfo

# ncview

$ ncview

# ncview <netcdf\_file>

$ ncview https://opendap.4tu.nl/thredds/dodsC/data2/pcrglobwb/version\_2019\_11\_beta/pcrglobwb2\_input/global\_30min/meteo/forcing/daily\_temperature\_cru\_era-interim\_1979\_to\_2010.nc

$ python

# inside python, please type import pcraster as pcr

1. **SCREEN (useful for multi-tasking in your linux terminal)**

**Introduction to screen**

Linux Screen allows you to:

1. Use multiple shell windows from a single SSH session (analogy: opening several tabs in your internet browser).
2. Keep a shell active even through network disruptions.
3. Disconnect and re-connect to a shell sessions from multiple locations.
4. Run a long running process without maintaining an active shell session.

A demo will be given by Edwin: Why screen? What happened if you don’t use it.

To start screen:

$ screen

To detach a screen

$ screen -d

To resume a screen

$ screen -r

You may have several screen sessions. To list all screen sessions that are currently active

$ screen –ls

There are screens on:

18666.pts-20.int1 (Detached)

734.pts-47.int1 (Attached)

2 Sockets in /var/run/screen/S-edwinvua.

To continue one of them: screen –dr <screen\_session\_name>

$ screen –r 18666.pts-20.int1

Within a screen session, you can also make multiple shell tabs/windows (RECOMMENDED).

Some shortcuts (Edwin’s favorite):

* 1. “ Ctrl-a ” ‘ c ’  creating a new window/tab

(at the same time, press “Ctrl” and “a”; then release both keys; then press “c”)

* 1. “ Ctrl-a ” ‘ n ’  go to the next window
  2. “ Ctrl-a ” ‘ p ’  go to the previous window
  3. “ Ctrl-a ” ‘ ” ’  show window list
  4. “ Ctrl-a ” ‘ A ’  provide a title for each window
  5. “ Ctrl-a ” ‘ d ’  detaching from screen
  6. “ Ctrl-a ” ‘ ? ‘  help

For more and better understanding, please see: <https://www.rackaid.com/blog/linux-screen-tutorial-and-how-to/>

Other links:

<http://aperiodic.net/screen/quick_reference>

<http://neophob.com/2007/04/gnu-screen-cheat-sheet/> (or google-ing: screen cheat sheet)

Problem with X11 $DISPLAY: If you reattach your screen, you may have to synchronize the variables $DISPLAY in your screen windows with the one in your main shell.

Before you reattach/resume your screen, check your current $DISPLAY variable in your main shell.

# still in your main shell

$ echo $DISPLAY

localhost:11.0

Resume your screen and check the current $DISPLAY in your screen window.

# still in your main shell

$ screen –r

# inside your screen session

$ echo $DISPLAY

localhost:10.0

If you try to visualize something, you may get an error message like this.

# inside your screen session

$ aguila

aguila: cannot connect to X server localhost:10.0

Here, you have to synchronize the $DISPLAY variable in your screen window to the $DISPLAY variable in your main shell.

# inside your screen session

$ export DISPLAY=localhost:11.0

Now, you should be able to visualize something (e.g. try aguila again).

**Exercise/homework for using screen:**

1. Start a screen session inside your linux machine (e.g. on velocity).
2. Inside your screen session, make three window shells and for every window shell, please echo and run the following lines:

# For window number 0.

$ echo “This is screen number 0”

$ top

# For window number 1.

$ echo “This is screen number 1”

$ echo $DISPLAY

# For window number 2.

$ echo “This is screen number 2”

$ ls -lah

# For window number 3.

$ echo “This is screen number 3”

$ mc

1. Detach your screen session and quit your ssh session from your linux machine (e.g. velocity).
2. Login again to your linux machine (e.g. velocity) and try to resume the previous screen. See if you can find and see your previous window sessions.