R – Exercises

Part 1. R and R package versions

1.1 Using Linux "module" function to switch to different versions of R

```
#Check version of default R
R --version

#List available R versions on BioHPC
module avail

#Switch to R version 3.5.0
module load R/3.5.0

#Check version of  R
R --version

#Restore default R
module unload R/3.5.0
```

1.2. Install a different version of R Package in your home directory

```
#Start R
R

#check session information
#pay attention to the BLAS library, which is  libRblas.so, not openBLAS.
sessionInfo()

#check version of gcc which is used to compile C/Fortran libraries
system('gcc -v')

#Check the version of an R package "testit"
packageVersion("testit")

#Check where "testit" is located
find.package("testit")

#Install an older version of "testit" in your home directory
PackageUrl <- "https://cran.r-
project.org/src/contrib/Archive/testit/testit_0.9.tar.gz"
install.packages(PackageUrl, repos=NULL, type="source")

#Check "testit" again
packageVersion("testit")
find.package("testit")

#Remove "testit" package installed by you
remove.packages("testit")

#Check "testit" again
packageVersion("testit")
```

```
find.package("testit")

#exit R
quit()
```

- This test shows that a package installed by yourself (located in your home directory) override the package installed by the system admin (located in "/programs/R-4.0.5/library").
- Here we use "remove.packages" function to delete a package. An alternative way is to delete the directory by this Linux command "rm -r ~/R/x86_64-pc-linux-gnu-library/4.0/testit"

Part 2. R-shell/R-script through Docker

2.1 Start a Docker container using the image "rocker/r-ver" (v4.1.1) built by the Rocker project

```
#When running this command, Docker would download the image from the Docker hub
and start a container
docker1 run -dit rocker/r-ver:4.1.1 /bin/bash

#get the Container ID, you will need this ID later
docker1 ps
```

2.2 Start an R-shell within container (replacing "xxxxxxx" with the container ID you get from last command)

```
docker1 exec -it xxxxxxx  R
```

Now you can run some R command. Remember, you are running as "root" user.

```
install.packages("testit")

find.package("testit")

library(testit)

find.package("testit")

quit()
```

2.3 Start a BASH-shell within container (replacing "xxxxxxx" with the container ID you get from "docker1 ps" command)

```
docker1 exec -it xxxxxxx  /bin/bash
```

Now you can run some BASH command. Again, you are the "root" user in container, you can do anything as "root".

```
pwd

ls -l

apt-get update

apt-get install nano
```

Access files of the host machine. The /workdir in the container is the same as /workdir/$USER of the host.

```
ls /workdir
```

Exit the container BASH

```
exit
```

2.4 Save the modified container to a new image file.

Now that you have installed some R packages and software into the container. You might want to save this container as a new image file. Otherwise if the container gets killed, all the newly installed software will be gone.

```
docker1 commit xxxxxxx myimage
```

Export the image as a file

```
docker1 save -o /workdir/$USER/myimage.tar biohpc_$USER/myimage
```

Once the file is saved, at another time and/or on another computer, you can load this image again.

```
#Now you remove all the containers created by you
docker1 clean all

#delete the image you just committed
docker1 rmi biohpc_$USER/myimage

#load the file into a new image
docker1 load -i /workdir/$USER/myimage.tar

#now you can start a new container with the loaded image
docker1 run -dit biohpc_$USER/myimage /bin/bash
```

2.5 Clean up

Check all containers and images on the host

```
#show images

docker1 images

#show containers

docker1 ps -a
```

Remove all the containers created by you. To remove only one container, do "docker1 stop xxxxxx" followed by "docker1 rm xxxxxxx".

```
docker1 clean all
```

Remove the images

```
docker1 rmi biohpc_$USER/myimage
```

If you have created some result files in the /workdir/$USER, these results files are owned by root. You cannot delete or move these files. To get ownership back, run

```
docker1 claim
```

This command would operate on the whole /workdir/$USER. If this directory is large, it could take a very long time. To claim a directory you can do "docker1 claim path_to_the_directory"

2.6 Run Rscript

So far, you have done interactive shell in Docker, including both R-shell and BASH sehll. If you just want to run a script, e.g. /workdir/$USER/myscript.R, you can do:

docker1 run --rm rocker/r-ver:4.1.1 Rscript /workdir/myscript.R

- The "--rm" option is to remove the container after the work is done.

Part 3. Rstudio through Docker

3.1 Start a Docker container R 4.1.1 using the image "rocker/rstudio" built by the Rocker project

```
#Pick a port number between 8009 and 8039 for Rstudio's web server.
#Check whether the port is available. Run the command below and if you see "8031
...  LISTEN", which means the port 8031 is taken by someone. Try a different
number

netstat -tulpn | grep 8031

#Start the docker container, replace 8031 with a number that is available. The
password is for Rstudio's built-in user "rstudio", you might want to change it
to something else

docker1 run -d -p 8031:8787 -e PASSWORD=12345 rocker/rstudio:4.1.1
```

3.2 Add your BioHPC user ID into the container

```
#get container id
docker1 ps

#add your BioHPC user id into the container (replace xxxxxxx with actual
container id)
docker1 exec xxxxxxx useradd -m -u `id -u` -d /home/$USER $USER

#set password for your user id in Container
docker1 exec -it xxxxxxx passwd $USER

#make you a sudo user in Container
docker1 exec xxxxxxx usermod -aG sudo $USER
```

3.3. Open your browser, and point to the Rstudio server you have started. Replace "cbsuxxxxxxx" with your server name, and "yyyy" with your port number.

http://cbsuxxxxxxx.biohpc.cornell.edu:yyyy

After you login with your BioHPC user id, change the working directory to "/workdir", so that you can access your data files in "/workdir/$USER".

```
setwd("/workdir")
```

To access the Linux shell in Container, click the "Terminal" tab. Try to install a Linux software in Terminal

```
sudo apt update

sudo apt install nano
```

3.4 Commit the change in your container to a new image file

If you install any software/packages in the Docker container, you might want to commit the change into a new Docker image file. Otherwise, if a container is killed, or host machine is restarted, everything in the container is lost. To do that, you need to go back to the host Linux shell, and run the command "docker1 commit". The "xxxxxxx" is the Docker container id.

```
docker1 commit xxxxxxx  myNewImageName

#now list the images on your server
docker1 images
```