Knowledge Book

Sonia García-Ruiz (s.ruiz@ucl.ac.uk)

2021-08-17

Contents

Chapter 1

Analysis of Genetic Association Studies

The purpose of this section is to provide guidance on performing a GWAS analysis step-by step, as well as introducing to useful software for undertaking a GWAS.

1.1 Useful Linux commands

First, I'll show you useful Linux commands. Linux is an operating system very well suited to dealing with large datasets.

- Create a shell script: touch script.sh

Chapter 2

GitHub

2.1 Merge

To merge two branches presenting conflicts, it is possible to indicate which changes are preferred by using a –ours/–theirs flag:

First, we checkout to the branch that we want to apply/commit the merge:

```
> git checkout testing
> git log --oneline
```

Merging would generate the following error:

```
> git merge testchanges2
warning: Cannot merge binary files: obj/Debug/netcoreapp2.1/CoExp_Web.csprojAssemblyReference.cach
warning: Cannot merge binary files: obj/Debug/netcoreapp2.1/CoExp_Web.Views.pdb (HEAD vs. testchang
warning: Cannot merge binary files: obj/Debug/netcoreapp2.1/CoExp_Web.Views.dll (HEAD vs. testchang
warning: Cannot merge binary files: .vs/CoExp_Web/v16/.suo (HEAD vs. testchanges2)
Auto-merging obj/Debug/netcoreapp2.1/CoExp_Web.csprojAssemblyReference.cache
CONFLICT (content): Merge conflict in obj/Debug/netcoreapp2.1/CoExp_Web.csprojAssemblyReference.ca
Auto-merging obj/Debug/netcoreapp2.1/CoExp_Web.Views.pdb
CONFLICT (content): Merge conflict in obj/Debug/netcoreapp2.1/CoExp_Web.Views.pdb
Auto-merging obj/Debug/netcoreapp2.1/CoExp_Web.Views.dll
CONFLICT (content): Merge conflict in obj/Debug/netcoreapp2.1/CoExp_Web.Views.dll
Auto-merging libman.json
CONFLICT (content): Merge conflict in libman.json
Auto-merging Views/Shared/_Layout.cshtml
CONFLICT (content): Merge conflict in Views/Shared/_Layout.cshtml
Auto-merging Views/Run/Help_Introduction.cshtml
Auto-merging Views/Run/Help_Catalogue.cshtml
Auto-merging Views/Run/Help_Annotation.cshtml
```

```
Auto-merging Views/Run/Help.cshtml
Auto-merging Views/Run/About.cshtml
Auto-merging .vs/CoExp_Web/v16/.suo
CONFLICT (content): Merge conflict in .vs/CoExp_Web/v16/.suo
Automatic merge failed; fix conflicts and then commit the result.
```

To obtain the detail of the files that present a conflict:

```
> git log --merge
> git log --merge
commit 29cd34cc267a858fe696e86981e83bd3af1abb85 (testchanges2)
Author: Sonia Garcia <43370296+SoniaRuiz@users.noreply.github.com>
Date:   Tue Nov 24 09:03:14 2020 +0000

    commiting .suo file in testchanges2

commit 7319379b1038dc5415cf035f915d91096c70af2f (origin/testchanges2)
Author: Sonia Garcia <43370296+SoniaRuiz@users.noreply.github.com>
Date:   Tue Nov 24 00:47:04 2020 +0000

    remove unnecessary libraries

commit 5bf600b6f5f1b41b42faaca9173f33efac4958f4
Author: Sonia Garcia <43370296+SoniaRuiz@users.noreply.github.com>
Date:   Mon Nov 23 23:49:54 2020 +0000

    new built

commit 1f60b3ffb5b558849fa061efd7af1796380de30c
Author: Sonia Garcia <43370296+SoniaRuiz@users.noreply.github.com>
Date:   Mon Nov 23 23:20:51 2020 +0000

    Update .suo

    commit suo

commit d1834c7ed4a83a95e08a148c7945525e3521981c
Author: Sonia Garcia <43370296+SoniaRuiz@users.noreply.github.com>
Date:   Mon Nov 23 23:10:18 2020 +0000

> git merge --no-ff testchanges2
error: Merging is not possible because you have unmerged files.
hint: Fix them up in the work tree, and then use 'git add/rm <file>'
hint: as appropriate to mark resolution and make a commit.
fatal: Exiting because of an unresolved conflict.
```

As we want to apply the comming changes, we use the flag “–theirs”, and the paths of the 7 conflictive files are updated:

```
> git checkout --theirs ./*
Updated 7 paths from the index
```

Then, we commit the changes of those 7 files and push them to the current repository (the ‘testing’ repository):

```
> git branch
  master
  testchanges
  testchanges2
* testing
> git add ./*
> git commit -m "checkout --theirs testing"
[testing bcc3b5c] checkout --theirs testing
> git push
...
> git status
On branch testing
Your branch is up to date with 'origin/testing'.

nothing to commit, working tree clean
```

Now, it’s time to checkout to the repository we want to merge from and apply the merge:

```
> git checkout testchanges2
Switched to branch 'testchanges2'
Your branch is ahead of 'origin/testchanges2' by 1 commit.
  (use "git push" to publish your local commits)

> git merge testing
Updating 29cd34c..bcc3b5c
Fast-forward
 Program.cs                         |  2 +-
 Properties/launchSettings.json     |  4 ++--
 Views/Run/About.cshtml             | 18 +++++++++---------
 Views/Run/Help.cshtml              |  4 ++--
 Views/Run/Help_Annotation.cshtml   |  4 ++--
 Views/Run/Help_Catalogue.cshtml    |  4 ++--
 Views/Run/Help_Introduction.cshtml |  4 ++--
 7 files changed, 20 insertions(+), 20 deletions(-)

> git checkout testing
Switched to branch 'testing'
Your branch is up to date with 'origin/testing'.
```

```
> git merge testchanges2
Already up to date.
```

Once the merge is done, we can apply the commit to the 'testing' repository and push the changes. After that, both repositories ('testchanges2' and 'testing') could be considered as merged, being the final result of their merge stored within the 'testing' repository:

```
> git add ./*

> git commit -m "testing version with the plot"
[testing 7c2f7dd] testing version with the plot
 18 files changed, 90 deletions(-)
 ...

> git push origin testing
 ...
```

2.1.1 Troubleshooting pages of interest

- This
- Merge a branch in Git
- Meaning of 'ours' and 'their' in Git.)

2.2 Stash

When we are working on our GitHub project, we may want to backup some of our changes. However, it is possible that those changes are not stable yet, or we think they are not ready to be committed into our repository. The stash command creates a backup copy of our changes, and keeps it ready to be committed into the repo once we feel it is ready.

It might happen that, once we think the stash is ready to be committed, GitHub doesn't allow us to do so because there are some 'untracked' files on it.

The only way I have found to deal with this issue, is to upload my stash into a brand new repo, and finally merge the new repo with the one I was intending to commit the changes to originally.

Chapter 3

Docker

3.1 Introduction

Nowadays, almost all companies and scientific labs which work with data are also becoming software producers. In some cases, the main reason for this software development is to publish effective scientific work. In other cases, its final aim might be just putting together a set of related functions that work for a similar final objective. However, in both cases, the final software product becomes highly shareable.

This characteristic, the shareability of the software produced, might seem very straightforward but in reality, it can turn into a scaring daunting task to be solved. Why? Because of the heterogeneity of all different platforms, operating systems, dependencies, versioning and so on that our software product makes use of. Please, be aware that prior to its first release, our software product might have been tested in a limited number of PC stations with determined characteristics that can be extremely different from any other potential user's PC station around the world.

All these reasons make Docker Platform really interesting and useful for both software producers and consumers.

3.1.1 What is docker?

Docker is a software platform created in 2013 by Docker, Inc. which its main objective is to *build, share, and run any app anywhere*, independently of the platform and environment where it is executed.

But what does this definition mean? It means that you admiringly can forget about dependencies, libraries, compatibility, etc. to make the app run correctly.

To some extent, you could think Docker as a *black box*, as a **snapshot** of the developer's laptop where she or he developt the software you are about to make use of. A snapshot of the precise moment when the developer decided to release the software. Thus, everything you need to run the app is already there, installed and configured inside the Docker object, ready to be used, with no compatibility issues at all.

3.1.2 Images and containers

Working with Docker, there are two main concepts you are going to hear about constantly: images and containers. An **image** is the virtual file or template that contains the whole set of instructions to build a container. An image is the raw Docker file you will directly download from Docker Hub developer's repository to your local. On the other hand, a **container** is the executable object directly generated from the image. A container will be the virtual object that represents the snapshot of the app developer's laptop.

In summary, the image is the virtual file that contains the raw instructions to build the executable app, and the executable app is the container itself.

3.1.3 Docker Hub

Docker Hub is an online platform that allows creating individual Docker repositories. A Docker repository is a personal account space where you can store one or more versions of the same Docker image, which are represented by tags.

Let us focus on the following image obtained from the Ubuntu Docker Repository:

3.1.4 Useful Commands

This page contains a list with some of the most common commands of Docker.

To download a Docker image from Docker Hub:

```
$ sudo docker push repository/name:tag
```

To run the image *name:tag*. The flag –**rm** indicates to remove the container after stopping the image; whereas the flag **-p** indicates the port on which we want to expose the execution of the image:

```
$ sudo docker run --rm -p 8500:80 name:tag
```

To list all docker images that are available in our local:

```
$ sudo docker images
```

To remove the image *image_name*:

```
$ sudo docker image rm image_name
```

To remove all orphaned images:

```
$ sudo docker rmi $(sudo docker images -f dangling=true -q)
```

To list all current containers:

```
$ sudo docker ps
```

To list all stopped containers:

```
$ sudo docker ps -a
```

To remove all orphaned containers:

```
$ sudo docker rm $(sudo docker ps -a -q)
```

To enter inside a container in execution:

```
$ sudo docker exec -it name_container /bin/sh
```

3.2 CoExp

This tutorial contains the instructions to install a local version of the CoExp Webpage by making use of the Docker technology. All the commands shown in this document have been tested in a Ubuntu18.04 machine.

3.2.1 Software requirements

Before downloading the CoExp Docker images, we first need to prepare the environment for the correct execution of Docker.

Thus, let's download/fetch new versions of the packages that are already installed in our Linux machine (all these commands have been tested in a Ubuntu18.04 machine):

```
$ sudo apt update
$ sudo apt upgrade
```

curl is a tool used to transfer data. We will make use of it later when we download the CoExp Docker images. To install *curl* in the machine:

```
$ sudo apt install curl
$ sudo curl --version
```

Now, we are ready to install the Docker technology in the machine. So, let's download it:

```
$ sudo apt install docker.io
```

Once the Docker installation has finished, we can enable and start it. The last instruction *sudo docker –version* will return the current Docker version installed:

```
$ sudo systemctl start docker
$ sudo systemctl enable docker
$ sudo docker --version
```

Finally, we need to install Docker-compose (more info here). Docker-compose is a brach of the Docker technology, which allows communicating different Docker images between them:

```
$ sudo curl -L https://github.com/docker/compose/releases/download/1.21.2/docker-compose
$ sudo chmod +x /usr/local/bin/docker-compose
$ docker-compose --version
```

3.2.2 Download Docker images of CoExp

If everything has gone as expected, the system should now be ready to download the two CoExp Docker images. There are two different images because one contains the user interface (UI) of the CoExp webpage, and the other one contains the backend of the CoExp WebPage (author Juan Botia).

In terms of the back-end of the CoExp Webpage, there are two different docker images available:

1. Complete version: this docker image contains the totality of all CoExp networks, and it is about ~4.5GB of size.
2. Lite version: this smaller docker image contains only the ROSMAP co-expression network. This image is about ~1.3GB of size.

Depending on which image you are interested in, the commands to execute are:

To download the complete version:

```
$ sudo docker pull soniaruiz/coexp:r
```

To download the lite version:

```
$ sudo docker pull soniaruiz/coexp:r-lite
```

3.2.3 Use Docker-Composer to build the images

The next step is to make the comunication between the two docker images possible. For that purpose, we need to download this **docker-compose.yml** file (in case you have opted by the complete backend version), or this **docker-compose.yml** file (if you have opted by the lite version). In any case, this file will make possible the correct communication between the two Docker images we downloaded in the previous step.

Additionally, the location of the downloaded **docker-compose.yml** file is not really important, but we recommend to place it in your **Home** folder (in case you are using a Linux machine).

Once the download has finished, use the following command to execute the **docker-compose** file and, therefore, to run your own Docker CoExp webpage:

```
$ sudo docker-compose up
```

Finally, to test whether the execution has been correct, please type the following URL into the address bar:

```
http://localhost:8088/
```

If everything has gone as expected, you should now be able to visualize your dockerized version of the CoExp webpage in your browser. Congratulations!

3.2.4 Juan's tutorial

Suppose we want to generate local TOM (Topology Overlap Measure) modules from a specific network so we may, independently from the CoExpNets Web application or within the application, plot or use networks in graph mode. We can do it by creating their module TOMs and get the corresponding graph in terms of a connectivity matrix we can plot.

We will exemplify this by using the frontal cortex network from GTEx V6 package as follows.

We launch all package stuff so we can start working with those networks.

```
library(CoExpNets)
library(CoExpGTEx)
CoExpGTEx::initDb()

netf = getNetworkFromTissue(tissue="FCortex",
                            which.on="gtexv6",
                            only.file=T)
```

And now (we assume it is not generated yet) create the module-TOMs for the Frontal Cortex Network as follows. As we see,

```
netf
```

the beta value for that network is 9, it is in the name between the tissue and the `.it.50.rds` token in the file name.

```
getModuleTOMs(tissue="FCortex",
              beta=9,
              out.path="~/tmp/mytoms/",
              which.one="gtexv6")
```

And we can see all module-TOMs created now at the `~/tmp/mytoms/` folder

```
list.files("~/tmp/mytoms/",full.names = F,recursive = F)
```

And now we can get any module's connectivity matrix so we can represent a graph for the TOM

```
getModuleTOMGraph(tissue="FCortex",
                  which.one="gtexv6",
                  module="black",
                  topgenes=10,
                  out.path="~/tmp/mytoms/")
```

And there you have it.

Chapter 4

Machine Learning Concepts

4.1 Introduction

Have you ever asked yourself what is the difference between **Artificial Intelligence** and **Machine Learning**? What about between **supervised** and **unsupervised learning**? Well, that's not surprising at all because trying to find out the right answer within the huge ocean of information that is the Internet, can become a really daunting task.

In this post, we will try first to define the most widely used Machine Learning concepts and finally trying to give clarifying examples of each one of them to try to help in the understanding of their meanings.

4.2 Artificial Intelligence vs. Machine Learning

What is the difference between Artificial Intelligence and Machine Learning? **Artificial Intelligence** is the concept of machines being able to perform tasks in a way that we would consider "smart". **Machine Learning** however is the current application of AI, where we just give machines access to data and let them learn for themselves (source Forbes).

The Machine Learning concept comprises different techniques whereby it is possible to make machines learning from diverse sets of data. Among the most important ones, we can highlight **supervised learning** and **unsupervised learning**.

4.3 Supervised learning

When the training data - the data we want machines learning about - comprises not only the input vectors but also their corresponding target vectors.

4.3.1 Classification

Classification is a supervised learning method used when the target vectors consist of a finite number of discrete categories.

The *iris* dataset available in R, for instance, can be used in classification problems because it provides different input vectors (*“Sepal.Length”, “Sepal.Width”, “Petal.Length” and “Petal.With”*) and a target vector (*“Species”*) with a finite number of categories (*“setosa”, “versicolor” and “virginica”*).

```
> summary(iris)
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width          Species
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100   setosa    :50
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   versicolor:50
 Median :5.800   Median :3.000   Median :4.350   Median :1.300   virginica :50
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
```

4.3.2 Regression

Regression is also a supervised learning method but only used when the target vectors consist of one or more continuous variables.

The *longley* R dataset is an example of this type of data. It presents a collection of inputs vectors (*“GNP.deflator”, “GNP”, “Unemployed”, “Armed.Forces”, “Population”, “Year”*) and a numeric vector output (*“Employed”*).

```
> summary(longley)
  GNP.deflator         GNP          Unemployed     Armed.Forces     Population
 Min.   : 83.00   Min.   :234.3   Min.   :187.0   Min.   :145.6   Min.   :107.6   Min.
 1st Qu.: 94.53   1st Qu.:317.9   1st Qu.:234.8   1st Qu.:229.8   1st Qu.:111.8   1st Q
 Median :100.60   Median :381.4   Median :314.4   Median :271.8   Median :116.8   Media
 Mean   :101.68   Mean   :387.7   Mean   :319.3   Mean   :260.7   Mean   :117.4   Mean
 3rd Qu.:111.25   3rd Qu.:454.1   3rd Qu.:384.2   3rd Qu.:306.1   3rd Qu.:122.3   3rd Q
 Max.   :116.90   Max.   :554.9   Max.   :480.6   Max.   :359.4   Max.   :130.1   Max.
```