Whole Transcriptome Data Analysis

Francesca CORDERO

Computer Science Department
Univeristy of Turin

francesca.cordero@unito.it

Marco BECCUTI

Computer Science Department
Univeristy of Turin

marco.beccuti@unito.it

Verona, 26-29 November, 2019

Whole Transcriptome Data Analysis

Reproducibility and docker

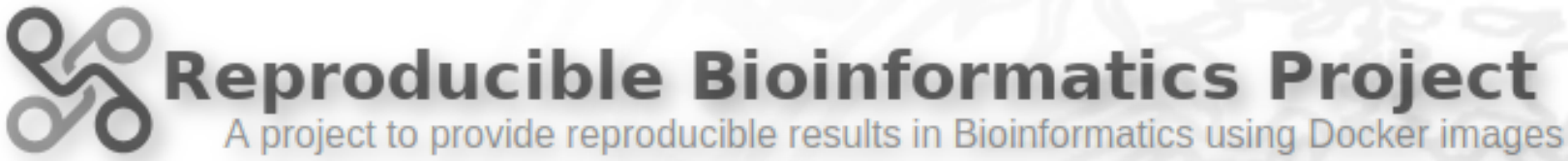
R - basic functions

Theoretical and practical session for RNA-seq analysis

Verona, 26-29 November, 2019

Reproducible research for NGS analysis

The package **docker4seq** was developed in **Reproducible Bioinformatics Project** to facilitate the use of computing demanding applications in the field of NGS data analysis.



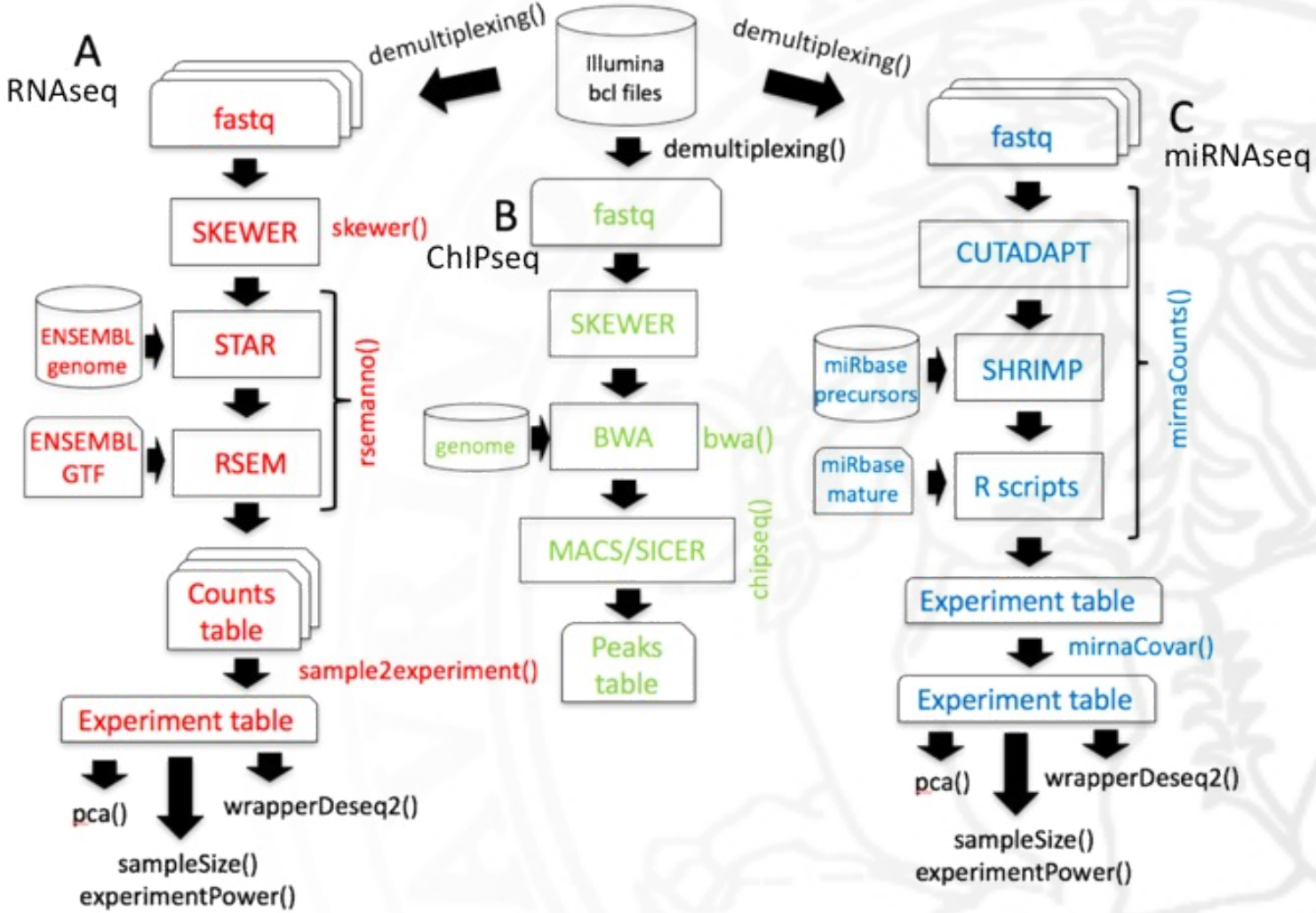
- Reproducible Bioinformatics Project (RBP) is community opens to anyone interested to shared workflows under the umbrella of reproducibility;
- To enable an easy access NGS data analysis pipeline to users without advanced computer science skills;
- To provide robust, reproducible, portable bioinformatics workflows.

http://reproducible-bioinformatics.org/

Reproducible research for NGS analysis

Current available workflows in RBP (docker4seq):

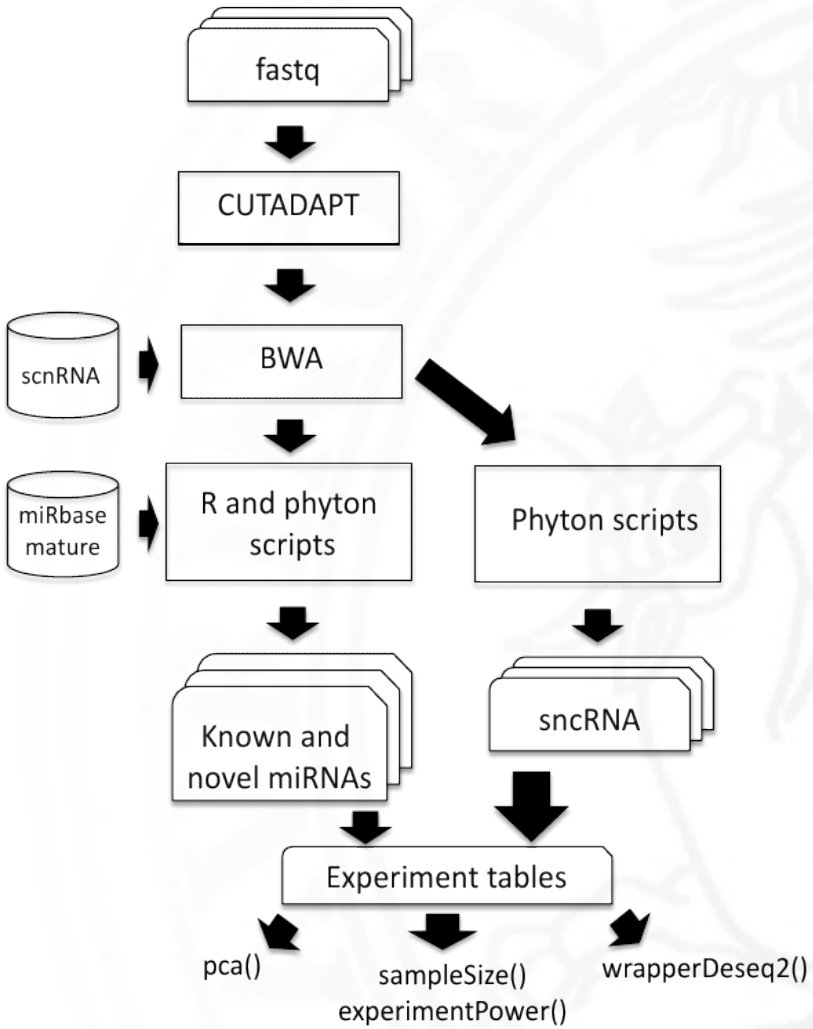
RNAseq, ChiPseq and miRNAseq analysis workflows



Reproducible research for NGS analysis

Current available workflows in RBP(docker4seq):

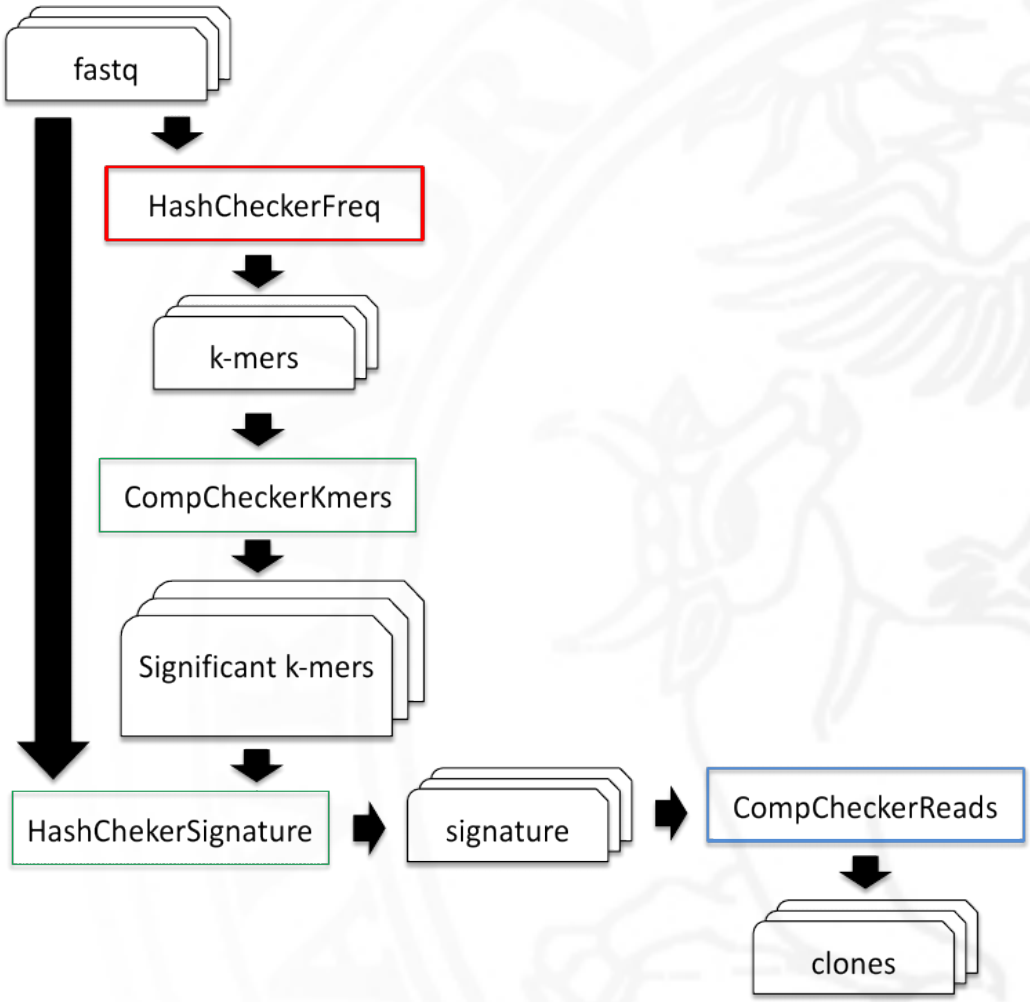
Small non-coding RNA workflow



Reproducible research for NGS analysis

Current available workflows in RBP(docker4seq):

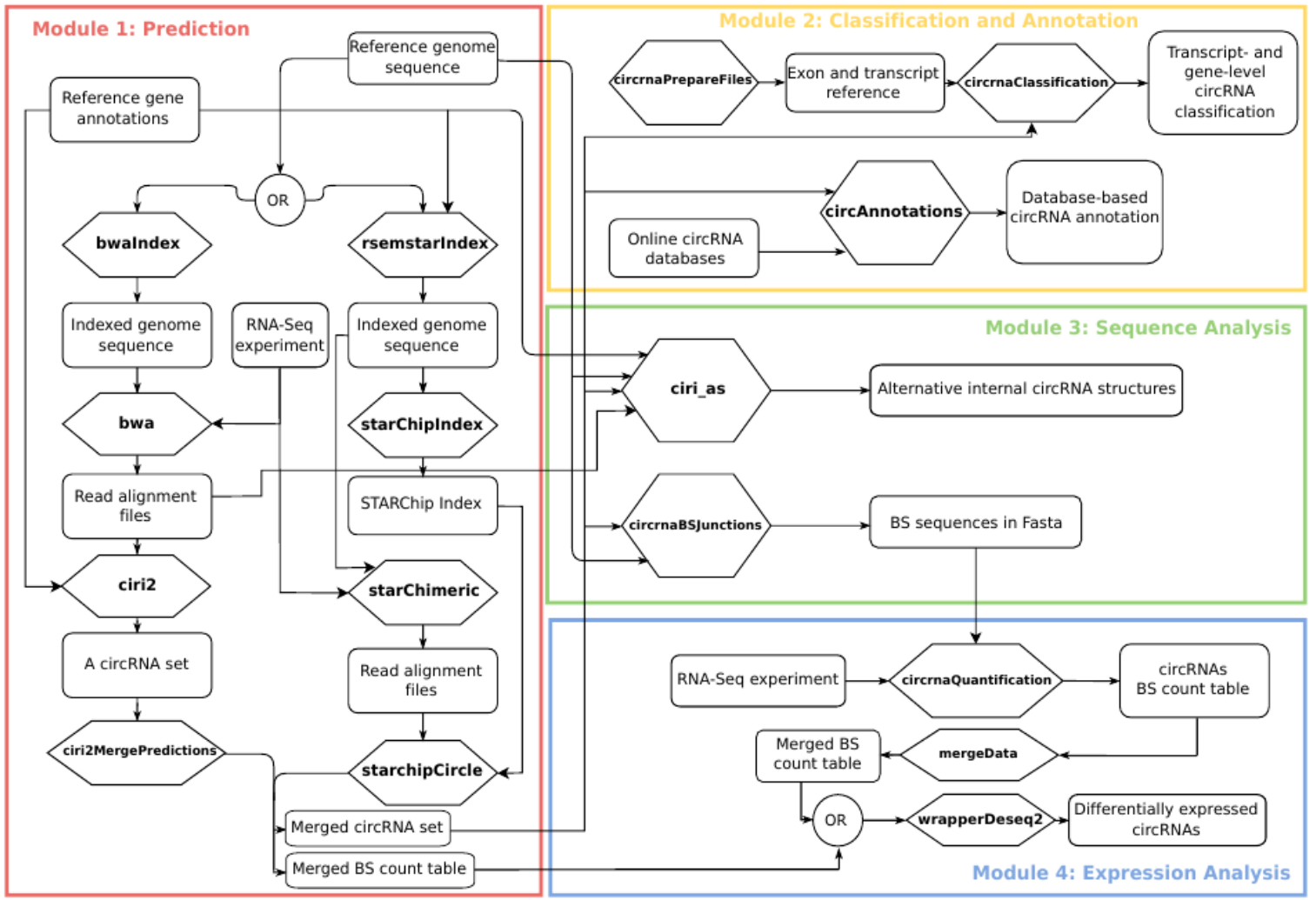
Minimal residual disease in B-cell lymphoma workflow



Reproducible research for NGS analysis

Current available workflows in RBP(docker4seq):

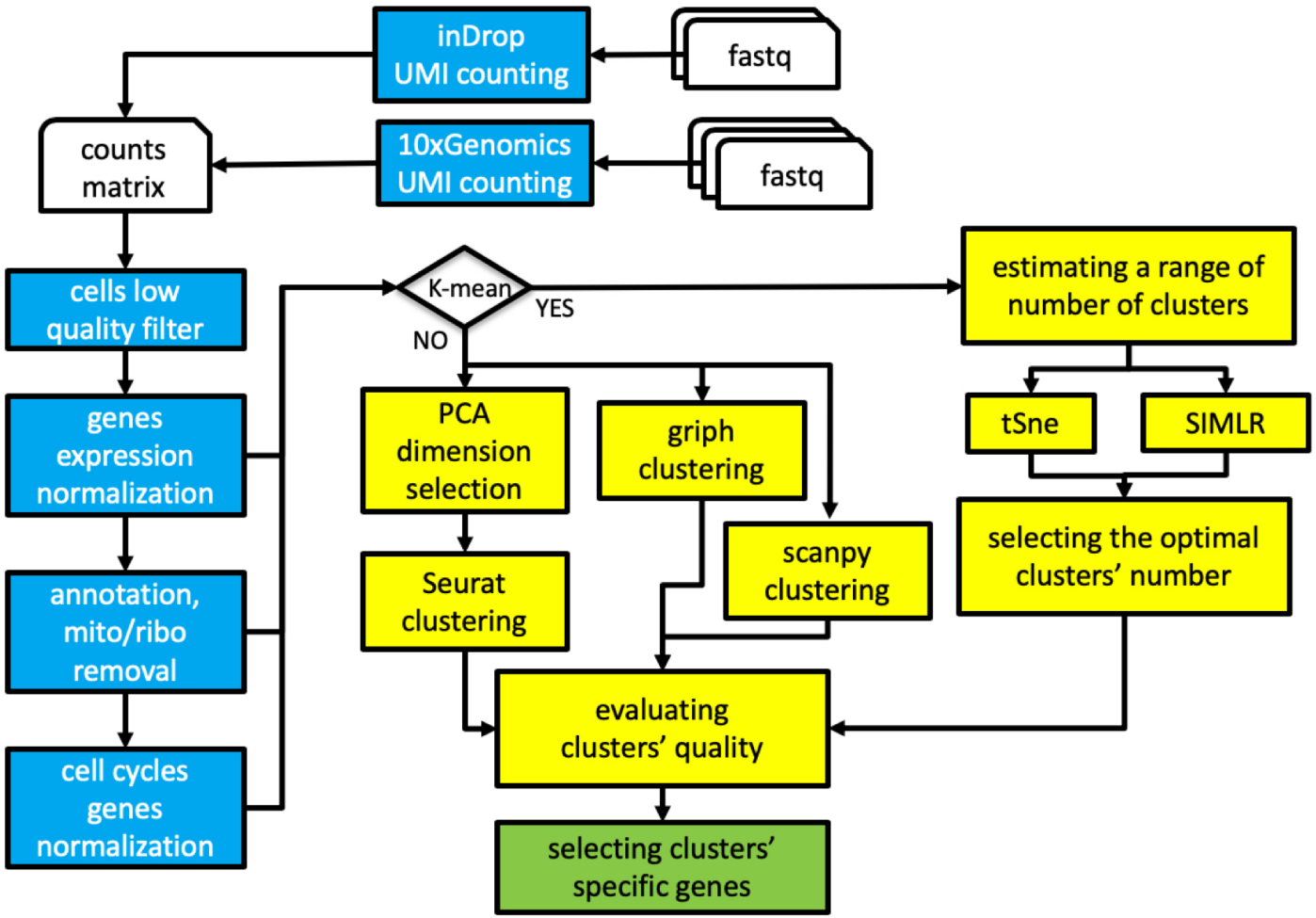
Prediction circular RNAs workflow



Reproducible research for NGS analysis

Current available workflows in RBP(rCasc):

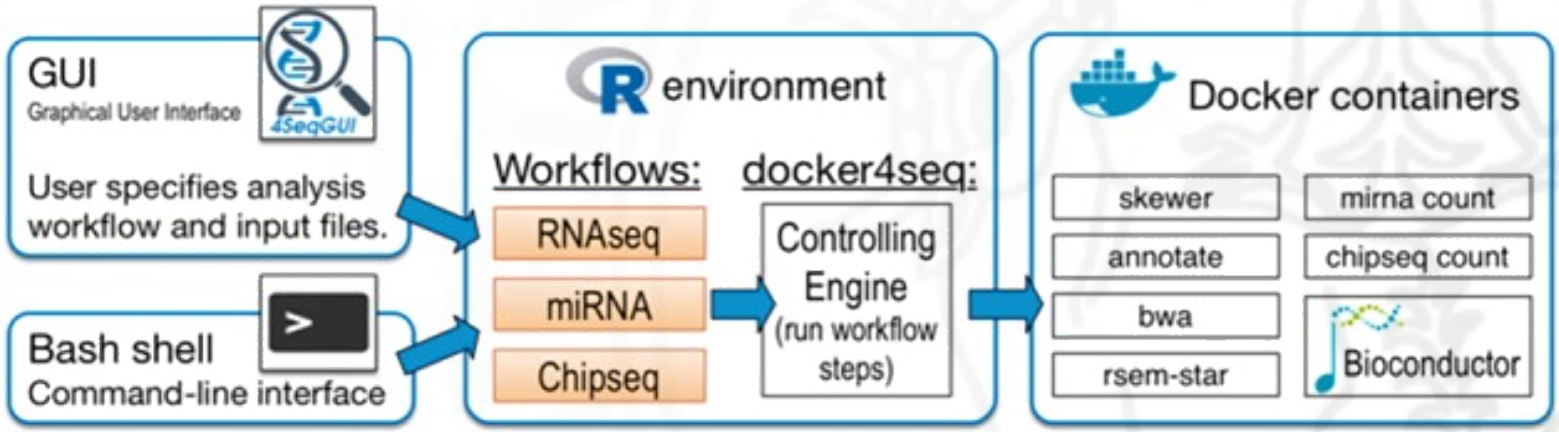
Single cell analysis workflow



Reproducible research for NGS analysis

The package **docker4seq** general schema:

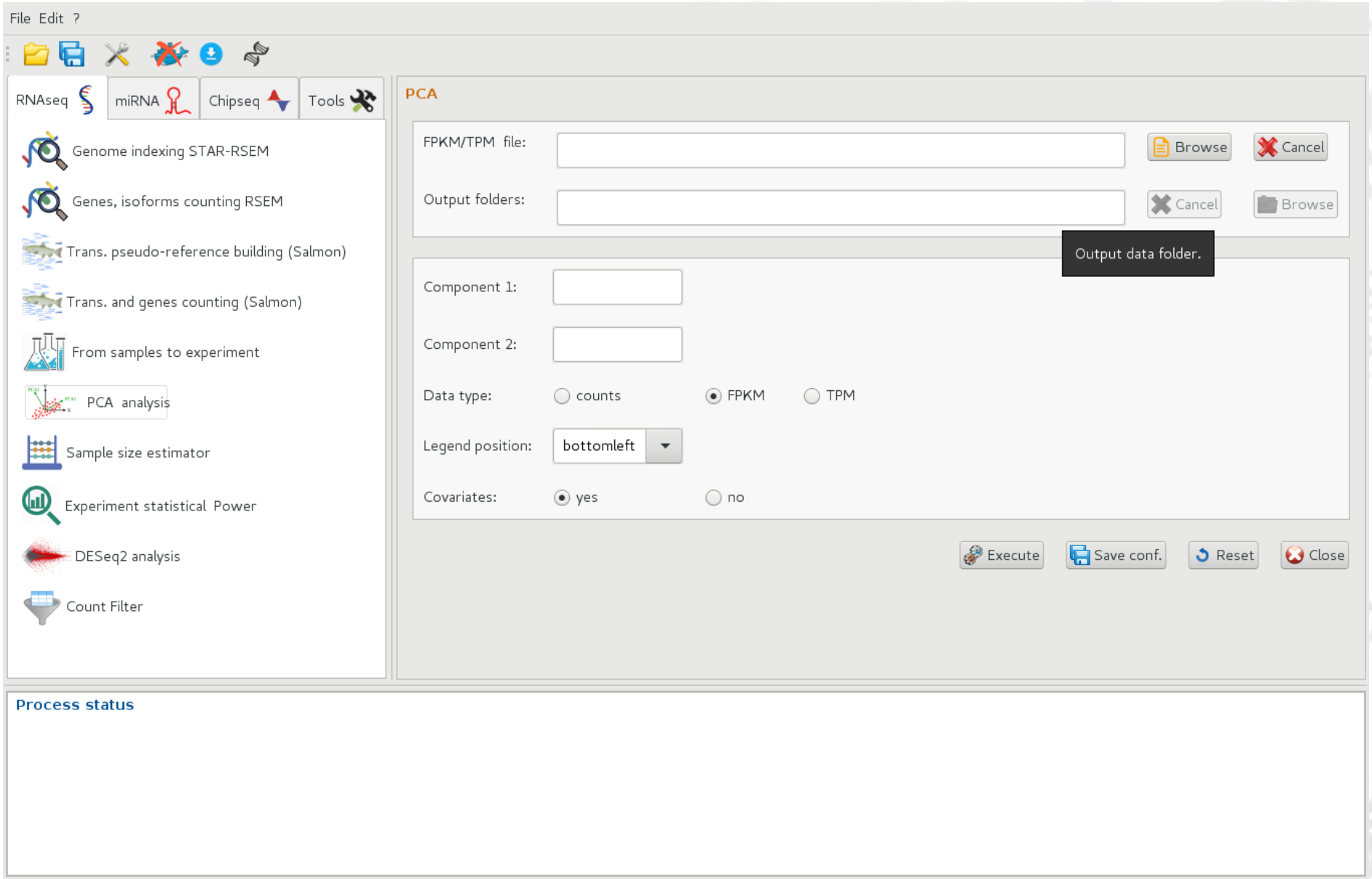
- Any workflow is specified as a set of **R functions** that defines and controls the correct execution of all its tasks;
- any single task is encapsulated into a **container** (i.e. docker images) to guarantee the computation reproducibility and portability.



Any workflow must be supported by an explanatory vignette

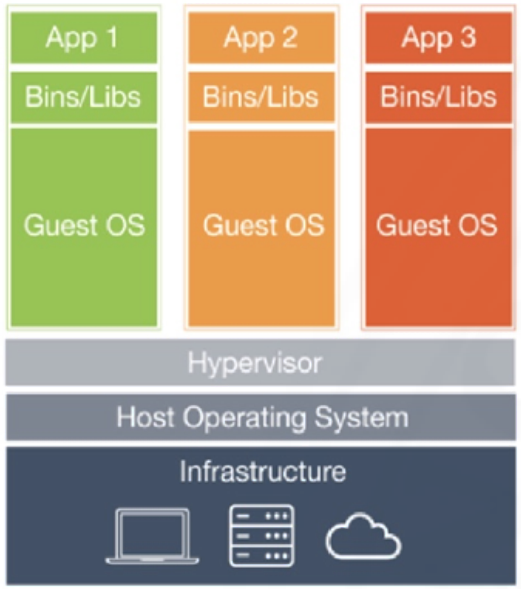
Reproducible research for NGS analysis

How to make easier the use of these workflows for beginner users:

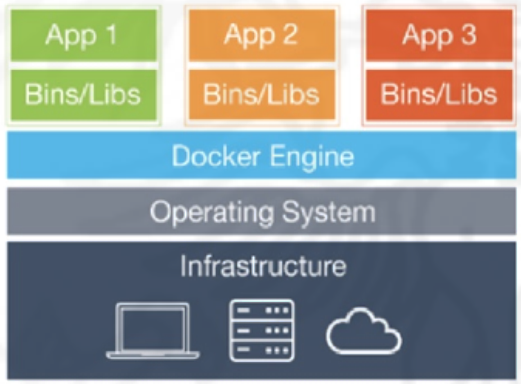


Reproducible research for NGS analysis

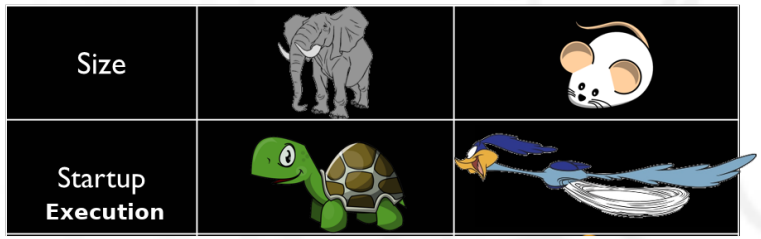
Container and Virtual machine are two virtualization techniques:



Virtual Machines



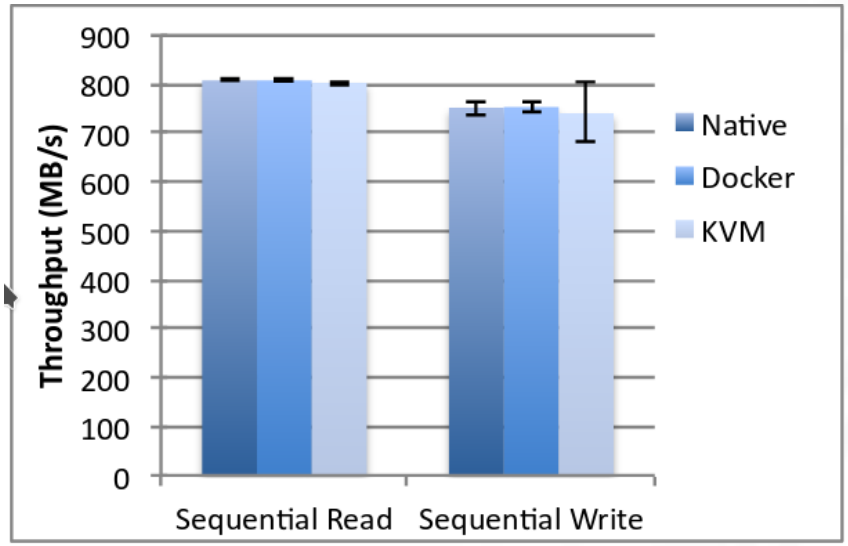
Containers

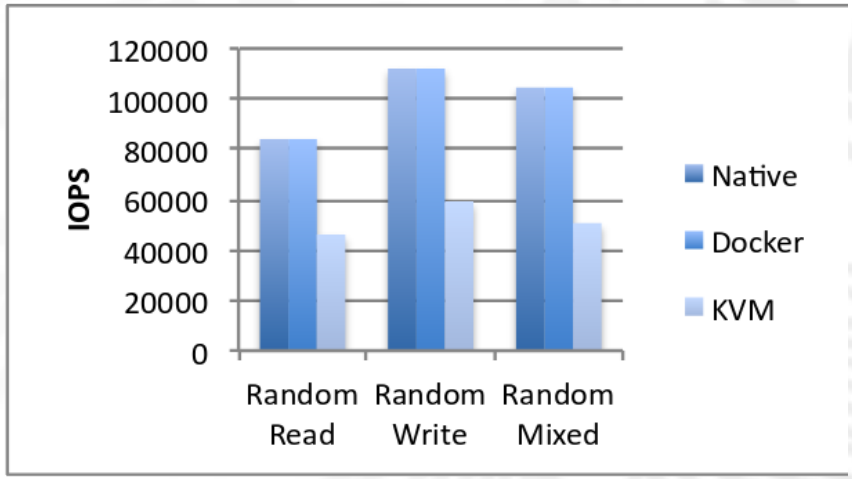


In Containers the sharing of same OS Kernel with the real machine reduces the portability, but ....

Docker container, VM and real server: a comparison

In [1] a comparison among physical server, KVM, and Docker is reported.





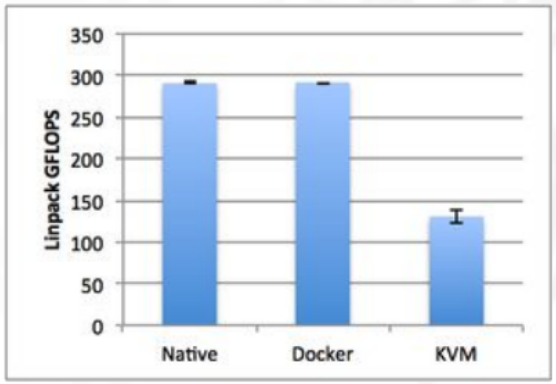
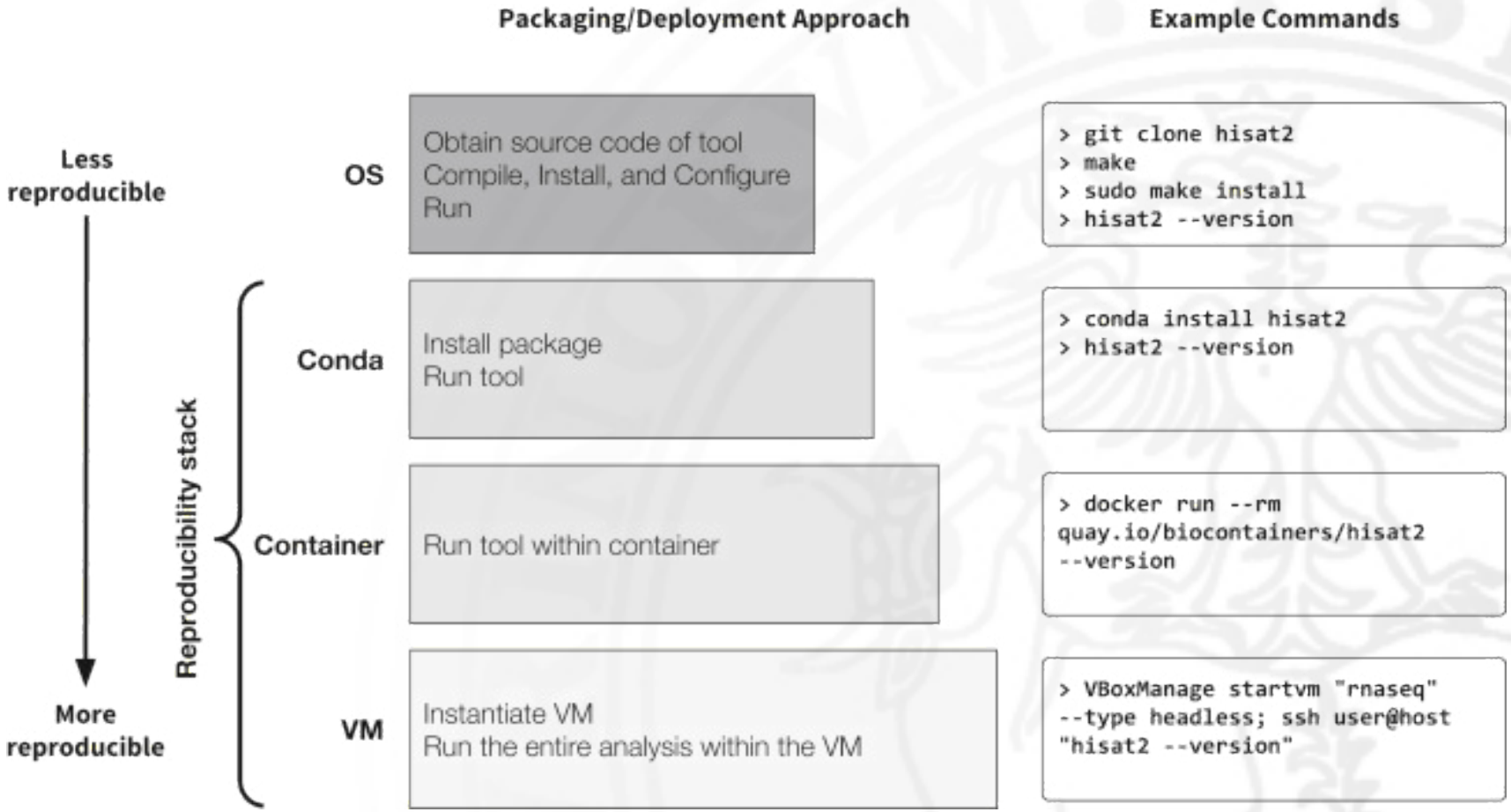


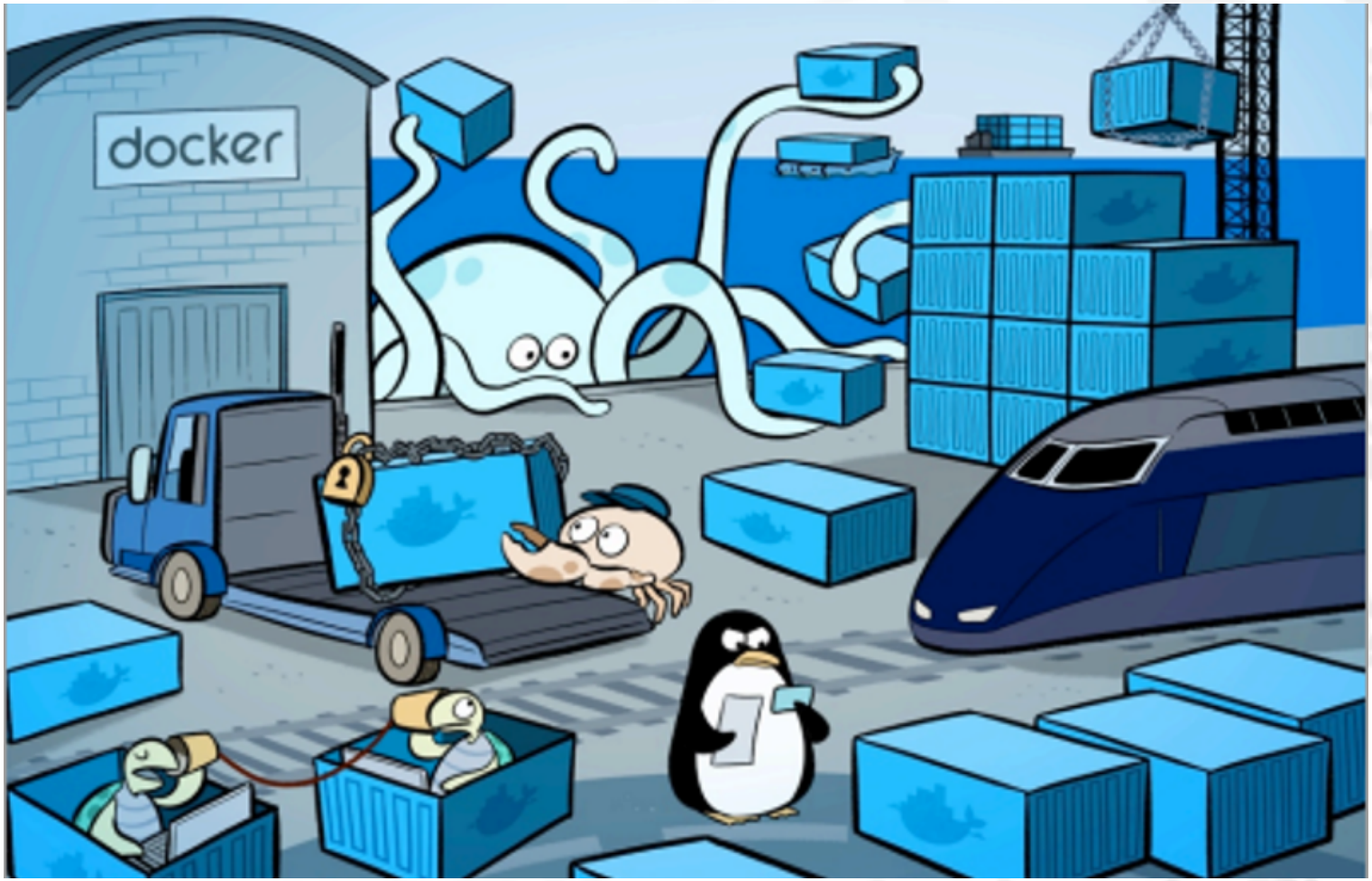
Figure 1. Linpack performance on two sockets (16 cores). Each data point is the arithmetic mean obtained from ten runs. Error bars indicate the standard deviation obtained over all runs.

[1] W. Felter, A. Ferreira, R. Rajamony and J. Rubio, ***An updated performance comparison of virtual machines and Linux containers***, 2015 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), Philadelphia, PA, 2015, pp. 171-172.

Computational Reproducibility Stack

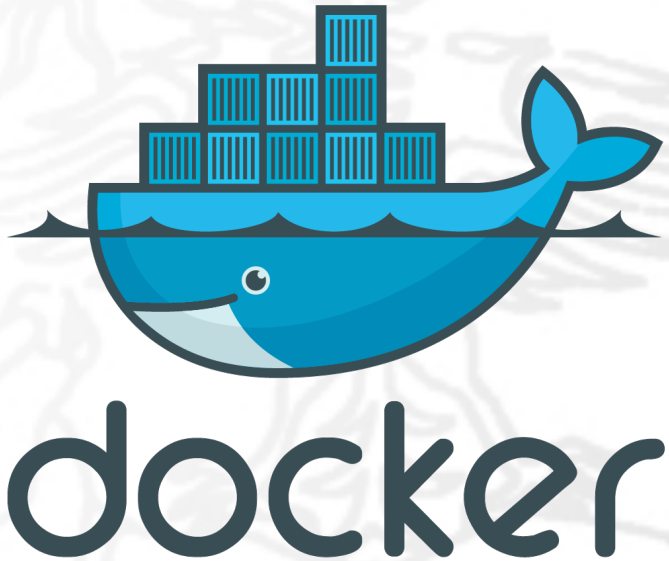


Docker project



Docker project

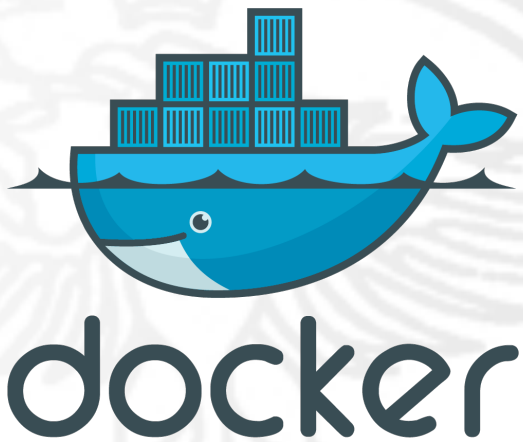
- Docker is an open-source project based on Linux containers.
- Others Linux container technologies include Solaris Zones, BSD jails, and LXC, which have been around for many years.



Why to use Docker??

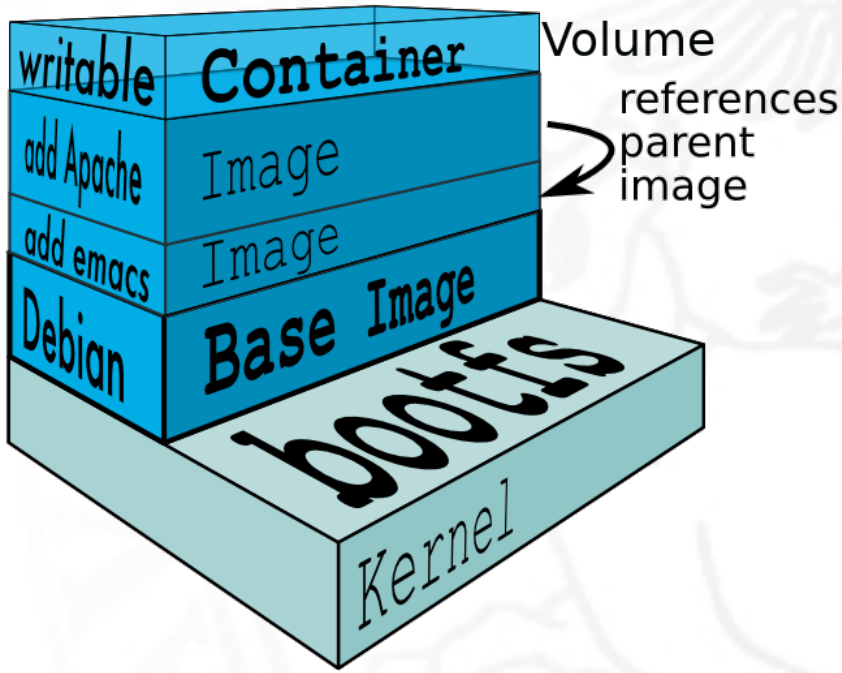
Why to use Docker?

- **Ease of use:** Docker has made it much easier for anyone to take advantage of containers in order to quickly build and test portable applications;
- **Speed:** Docker containers are very lightweight and fast.
- **Docker Hub:** Docker users also benefit from the increasingly rich repository of Docker Hub, which you can think of as an "app store for Docker images";
- **Modularity and Scalability:** Docker makes it easy to break out your application's functionality into individual containers.



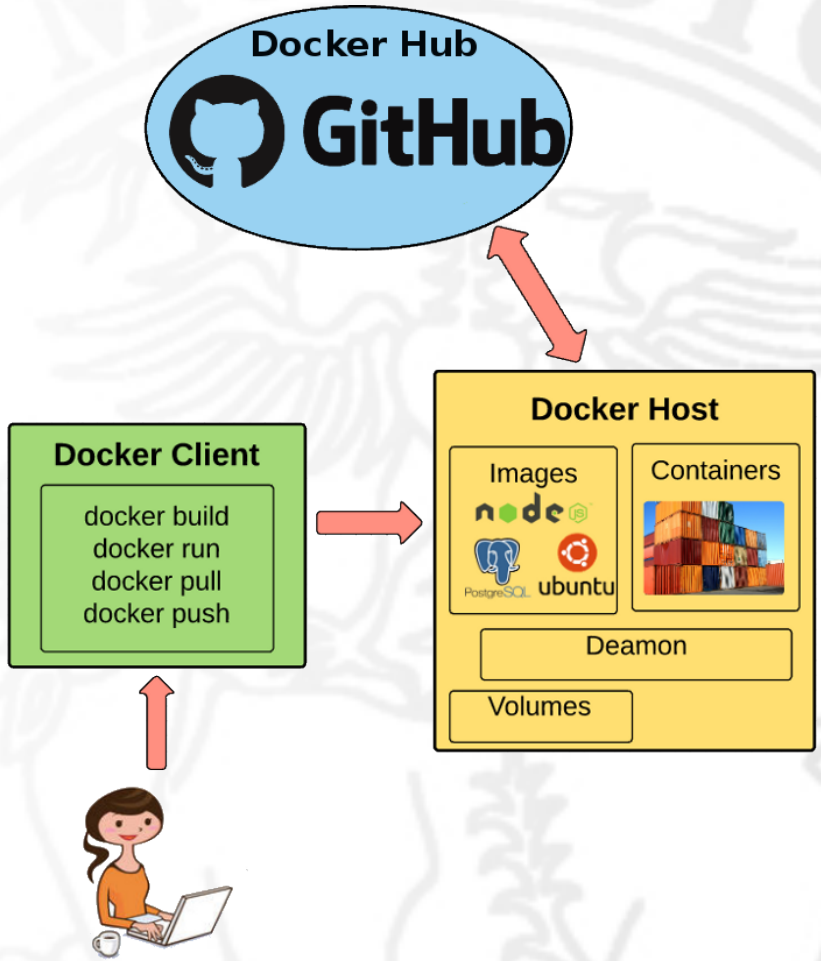
Docker basics

- ***Image*** is an executable package that includes everything needed to run an application (i.e. its code, libraries, environment variables, and configuration files);
- ***Container*** is a run-time instance of an image;
- ***Volume*** is used to share files from the host machine to containers.



Docker schema

- **Docker client:** provides an interface for users;
- **Docker Host:** executes the commands sent to the Docker Client;
- **Docker Hub:** remove repository storing docker's images.



How to install docker4seq packages

How to install docker4seq packages

- You need to install on your machine:
  - A Linux-based operating system (i.e Ferdora, Ubuntu, Idots);
  - R environment (see http://cran.mirror.garr.it/mirrors/CRAN/)
  - Docker environment (see https://www.docker.com/)
- The minimal hardware requirements are an x86_64 compatible processor, 32Gb RAM, an SSD 250GB.

How to install docker4seq packages

- You need to execute in R environment the following commands:

$$> install.packages(\text{“}devtools\text{”})$$

$$> library(devtools)$$

$$> install\_github(\text{“}kendomaniac/docker4seq\text{”}, ref = \text{”}master\text{”})$$

- To use and download docker images (about 50Gb):

$$> library(docker4seq)$$

$$> downloadContainers(group = \text{”}docker\text{”})$$

During this course ...

- We use Rstudio web server running on two remote machines in the HPC4IA cluster (see https://hpc4ai.it/)