Computational Practical 1: Computing environment and setup
Module Developers: Dr. Stanford Kwenda, George Githinji, Martin Aslett

1.0 Learning outcomes

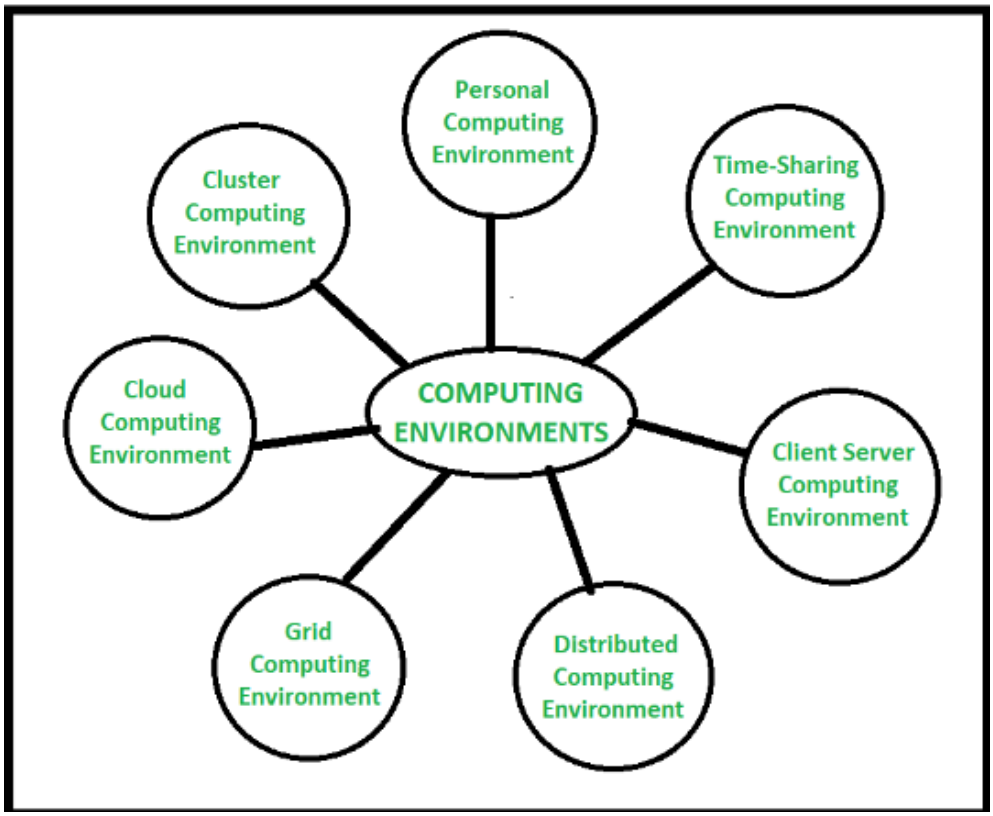
- Install bioinformatics tools successfully using the linux command-line interface
- Understand how to create various environments on linux-based computer systems and manage bioinformatics packages or tools.

1.1 Introduction

The field of bioinformatics relies heavily on Linux-based computers and software. In most set-ups users connect to a central server from their individual laptops in order to access bioinformatics tools and perform analysis. Often, such linux systems are maintained by an experienced system administrator, e.g. (remote) servers, high performance computing (HPC) clusters.

1.2 Computing environments

- Technology infrastructure + software platforms used to run applications
- Choice of environment depends on specific requirements and the financial resources available to purchase/pay for access



https://www.geeksforgeeks.org/computing-environments/

1.3 Package managers

- Essential tools for managing software installations, updates and dependencies
- Automated and simplified retrieval, configuration and installation of packages from repositories
- Some common package managers include:
    - APT (Advanced Package Tool) e.g. on Debian-based distributions (e.g. Ubuntu)
    - YUM (Yellowdog Updater, Modified) e.g. Red Hat based distros such as CentOS, Fedora

```
yum #Main command for package management
#Example commands:
sudo yum update #Updates installed packages
sudo yum install <package> #Installs a package
sudo yum remove <package>: #Removes a package
sudo yum search <keyword>: #Searches for packages
```

    - Conda

```
conda #Main command for package management
#Example commands:
conda update conda #Updates conda itself
conda install <package> #Installs a package
conda remove <package> #Removes a package
conda search <package> #Searches for packages
```

        - Bioconda channel - distribution of bioinformatics software packages

```
conda config --add channels defaults
conda config --add channels bioconda
conda config --add channels conda-forge
```

- It's generally straightforward to install software using package managers

1.3.1 Using APT e.g. on Ubuntu

```
stan@LT-Stanford:~$ apt
apt 2.0.10 (amd64)
Usage: apt [options] command

apt is a commandline package manager and provides commands for
searching and managing as well as querying information about packages.
It provides the same functionality as the specialized APT tools,
like apt-get and apt-cache, but enables options more suitable for
interactive use by default.

Most used commands:
  list - list packages based on package names
  search - search in package descriptions
  show - show package details
  install - install packages
  reinstall - reinstall packages
  remove - remove packages
  autoremove - Remove automatically all unused packages
  update - update list of available packages
  upgrade - upgrade the system by installing/upgrading packages
  full-upgrade - upgrade the system by removing/installing/upgrading packages
  edit-sources - edit the source information file
  satisfy - satisfy dependency strings
```

Exercise 1:

1. Update Package Information:

```
sudo apt update
```

2. Upgrade installed packages

```
sudo apt upgrade
```

Upgrade all installed packages to their latest versions

3. Install packages

```
sudo apt install <package>
```

4. Remove packages

```
sudo apt remove <package>
```

5. Auto-remove unused dependencies

```
sudo apt autoremove
```

1.4 What is sudo?

```
sudo #super user do!
su #switch user
```

su #su will ask for the root password and give you a super user prompt, signified by the # symbol

1.5 Manual installation of bioinformatics tools

1.5.1 Compiled binaries

1.5.2 Installing manually using make, configure, and make install

1.6 Environment management

Different bioinformatics tools may have conflicting dependencies or version requirements. This often complicates installation of bioinformatics tools. Thus, in bioinformatics there is often a need to create isolated environments which will allow specific versions of software tools and their dependencies to be installed without affecting the systems global configuration. Different approaches can be used to create such isolated environments and handle dependencies issues::

1. **Conda environments**

Managing software environments using conda is helpful in avoiding dependency conflicts and allows various bioinformatics tools to work together correctly. It makes sharing environments with others easier easy - allowing for reproducibility

2. **Modules**

3. **Containers**

Containerization technologies such as Singularity and Docker can be used to create isolatedEnvironments with all dependencies pre-installed to overcome dependency issues. Powerful tool for creating isolated and reproducible environments to run software and bioinformatics analyses. Most popular containerization software platforms in bioinformatics are:

- Docker
- Singularity

Main benefits include:

- Reproducibility
- Isolation
- Portability

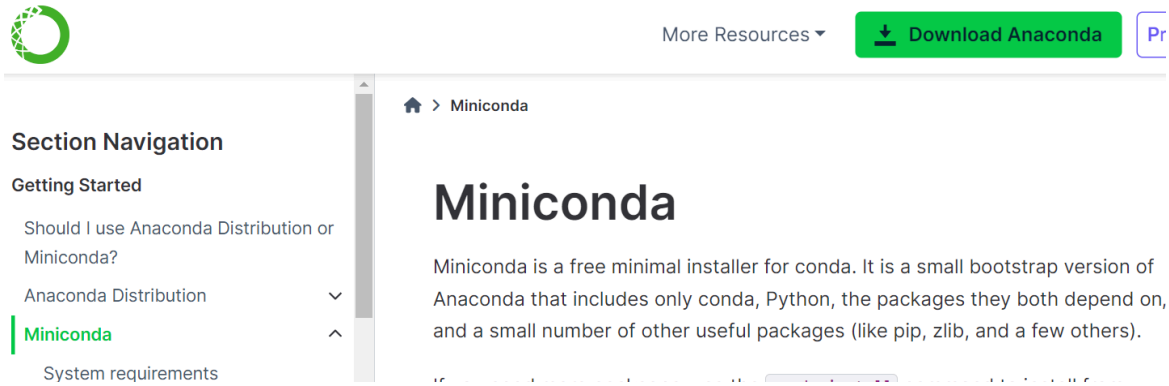
Cloud-based solutions

- Scalable, flexible and often cost-effective
- Virtualized environments that can be used to deploy bioinformatics software

In our practical session we will explore conda including how to create, activate, remove environments etc.

Exercise 2: Creating conda environments

1. First you will need to check/ensure that conda is already installed on your system
    a. Download the installer e.g. miniconda



https://docs.anaconda.com/miniconda/

2. Install miniconda

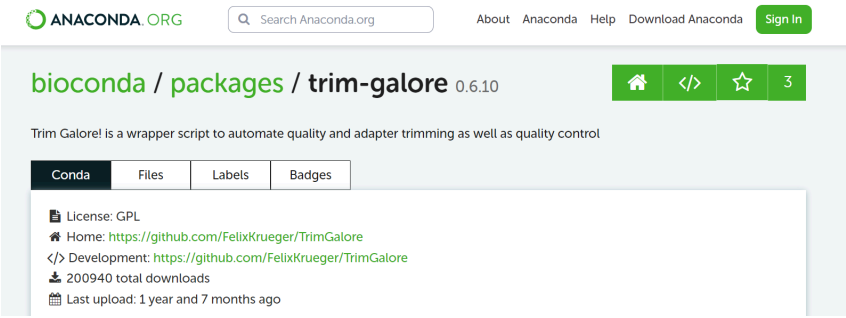
```
# download miniconda
cd $HOME
wget -c https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh

# install miniconda
bash Miniconda3-latest-Linux-x86_64.sh
```

3. Check/test if your installation was successful (You can use these commands to check if you already have conda installed before doing steps 1 & 2 above)

```
conda list
# OR
conda info —-env
```

4. Setting up your first env
Let's assume that you have just been given the task to perform some QC on a set of raw fastq files (short reads), and you have decided to perform the QC and filtering using e.g. a tool called trimGalore. Your task is to setup a readqc environment which will safely and correctly install the tool you have decided to use and the requisite dependencies

There are 2 ways of creating this env using conda, you can execute a one-liner command or create an “installation recipe file” (.yml) with the required tools and channels. We will try both approaches. But first we need to check the conda repo for the tool we are interested in to choose a specific version.



https://anaconda.org/

Do either of the following to create your environment:

a. Using a one-liner command, you can do the following

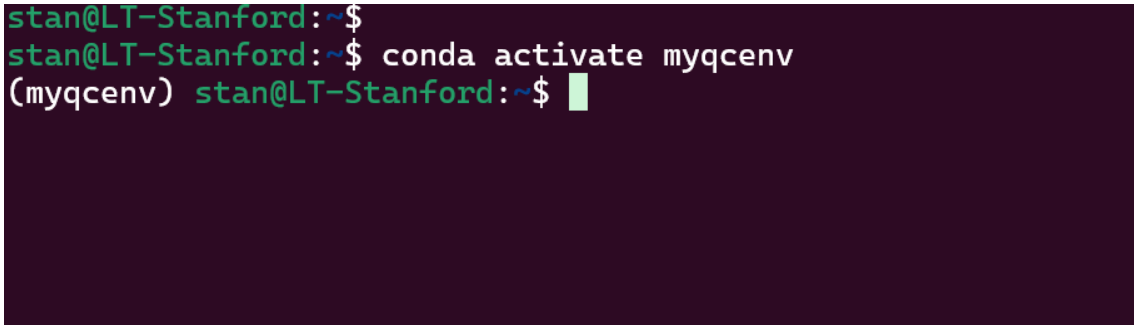
```
conda create -n mytest trim-galore==0.6.10 -c bioconda --solver=libmamba
```

b. Using an environment definition file

```
conda env create -f ~/cp1/qc_env_file.yml  --solver=libmamba
```

Once you have successfully created the env, the next step is to activate in order to use the tools/packages within this environment

```
conda activate myqcenv
```



Check if you now have trim-galore in your path

```
trim_galore --version
```

Deactivate an environment

```
conda deactivate
```

List conda environments currently installed

```
conda env list

# OR

conda info --env
```

List conda packages currently installed

```
conda list

# Or select packages starting with a specific prefix
conda list ^py

# for more options
conda list --help
```

Remove a conda environment

```
conda env remove -n mytest --solver=libmamba
```

Accessing a linux server from a Windows computer

1. Install WSL
   https://learn.microsoft.com/en-us/windows/wsl/install
   https://contabo.com/blog/how-to-install-wsl2-on-windows-10/
2. Install PuTTY
   a. https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html
   b. WinSCP - for securely copying files between the server and your local computer
      i. https://winscp.net/eng/download.php

Building bioinformatics workflows/pipelines

1. Bash
2. Python
3. Nextflow

Exercise 3: Setting-up a conda environment for QC tool that processes nanopore long read data

1. In this exercise we will be setting up an env for a tool called pycoQC
2. Using the steps outlined below, go ahead and create a new environment called pycoqc
   a. First search for the package name on Anaconda.org to get the latest version
   b. Create an environment definition file. It should have the following sections:
      i. You can create the file using nano on the linux terminal

```
name: pycoqc
channels:
  - defaults
  - bioconda
  - conda-forge
dependencies:
  - pycoqc=
```

Set the appropriate version

```
# Hints for using nano:
nano pycoqc_env.yml # To create a file:
Ctrl + O #To save a file
Ctrl + X # To exit
```

c. Create the env using `conda env create`
   If unsure check the commands options using `conda env create --help`
d. Activate the environment
e. Print out the version of pycoQC
f. Deactivate the environment and try to print out the version of pycoQC
g. Activate the pycoqc environment and show all the options for using pycoQC

Further reading:

1. https://ubuntu.com/server/docs/package-management
2. https://modules.sourceforge.net/
3. https://docs.sylabs.io/guides/3.0/user-guide/index.html
4. https://docs.conda.io/projects/conda/en/4.6.0/user-guide/index.html
5. https://a-slide.github.io/pycoQC/
6. https://github.com/FelixKrueger/TrimGalore