Managing software environments with

SciLifeLab

NBIS

Full reproducibility requires the possibility to recreate the system that was originally used to generate the results.

Conda is a package, dependency, and environment manager

Package: any type of program (e.g. bowtie2, snakemake etc.)

Dependency: other software required by a package

Environment: a distinct collection of packages

Conda keeps track of the dependencies between packages in each environment

Conda channels

Channels are remote directories containing packages.

Two common examples are

- bioconda: a channel specializing in bioinformatics software
- conda-forge: a community-led channel made up of thousands of contributors

Conda, Anaconda, Miniconda...

- Conda: the package manager itself, written in python
- Anaconda:
    - an installer for conda containing over 7,500 open-source packages
    - a cloud service where conda packages are hosted (anaconda.org)
    - a distribution of packages for data science (anaconda.com)
- Miniconda: an installer for conda containing only the most necessary packages to get started

Defining and sharing environments

Define a Conda environment in an `environment.yml` file:

```
channels:
  - conda-forge
  - bioconda
dependencies:
  - fastqc=0.11
  - sra-tools=2.8
  - snakemake=4.3.0
  - multiqc=1.3
  - bowtie2=2.3
  - samtools=1.6
  - htseq=0.9
  - graphviz=2.38.0
```

```
# Create a new environment from YAML
$ conda env create --name project_a -f environment.yml
```

```
# Update an existing environment from YAML
$ conda env update -f environment.yml
```

```
# Export existing environment as new YAML file (including all dependencies)
$ conda env export > environment-full.yml
```

```
# Export historical environment, i.e. packages listed in the original YAML
$ conda env export --from-history > environment-history.yml
```

Questions?

SciLifeLab

NBIS