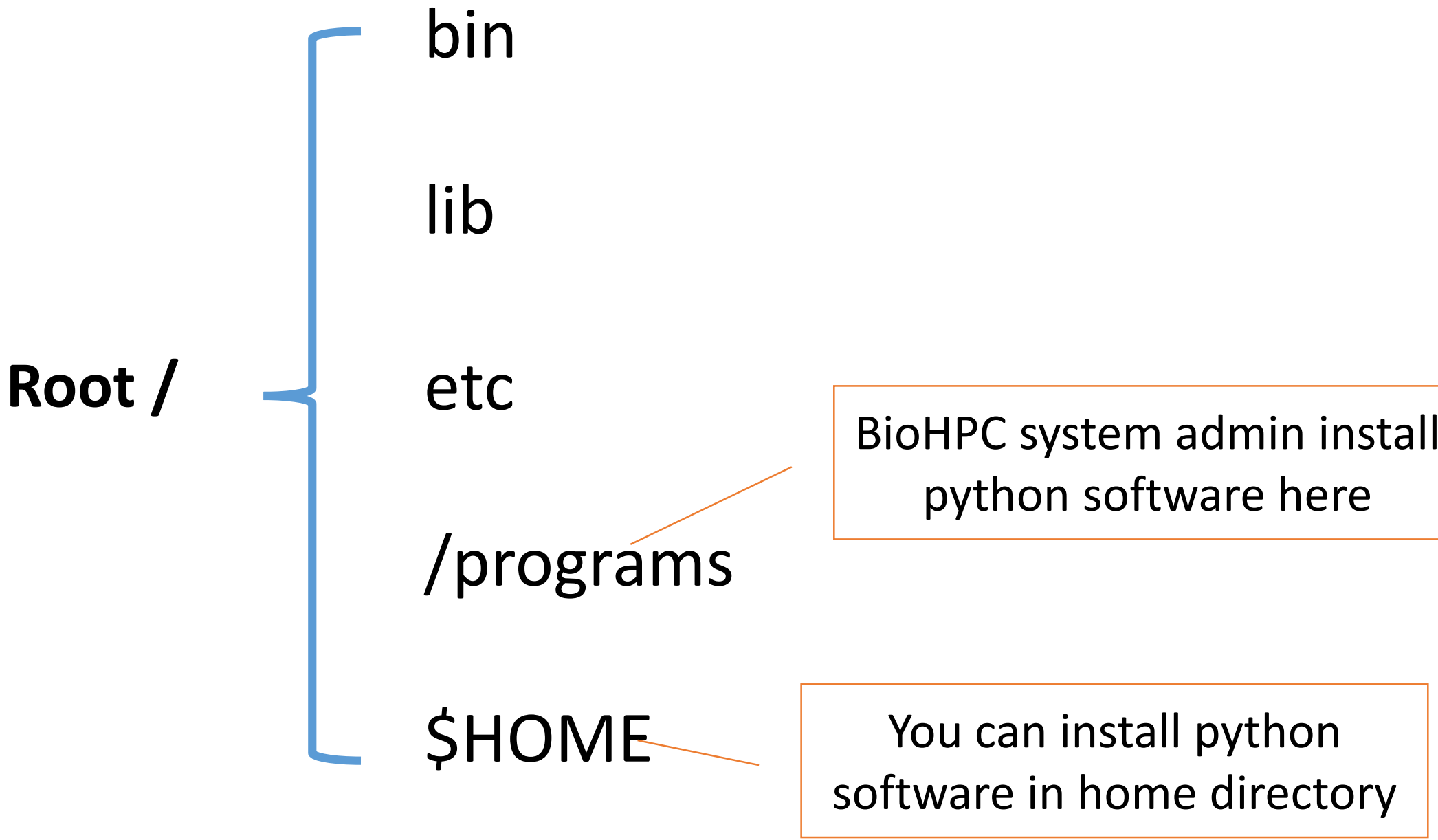
Linux Software Installation

Part 2

Qi Sun
Bioinformatics Facility


Root /
bin
lib
etc
/programs
$HOME
BioHPC system admin install python software here
You can install python software in home directory

How to install software by yourself?

Default, but only the root user has privilege to write.

/usr

/usr/local

You have write privilege.

/home/xxxxx

/workdir

How to install software by yourself?

- Conda
- Docker
- Change default installation directory.

Python

	PYTHON	PERL	R
Repository	**PYPI ***	**CPAN**	CRAN
Installation tool	**pip**	**cpan**	install.packages
Library PATH	**PYTHONPATH**	**PERL5LIB**	

* pronounced "pie pee eye"

PIP – A tool for installing/managing Python packages

- **PIP use PYPI repository to download software;**
- **Every python installation has it companion “pip”. (including python in Conda)**

  pip -> python2
  pip3 -> python3

Two versions of Python on BioHPC

python 2

Installation:

pip install myPackage

Run software:

python myscript.py

python 3

Installation:

pip3 install myPackage

Run software:

python3 myscript.py

Two ways to do “pip install” to your home directory

```
pip install deepTools --user
```

Installed in
$HOME/.local/bin
$HOME/.local/lib & lib64

* Suitable for personal installation

```
pip install deepTools \
--install-option="--prefix=mydir" \
--ignore-installed
```

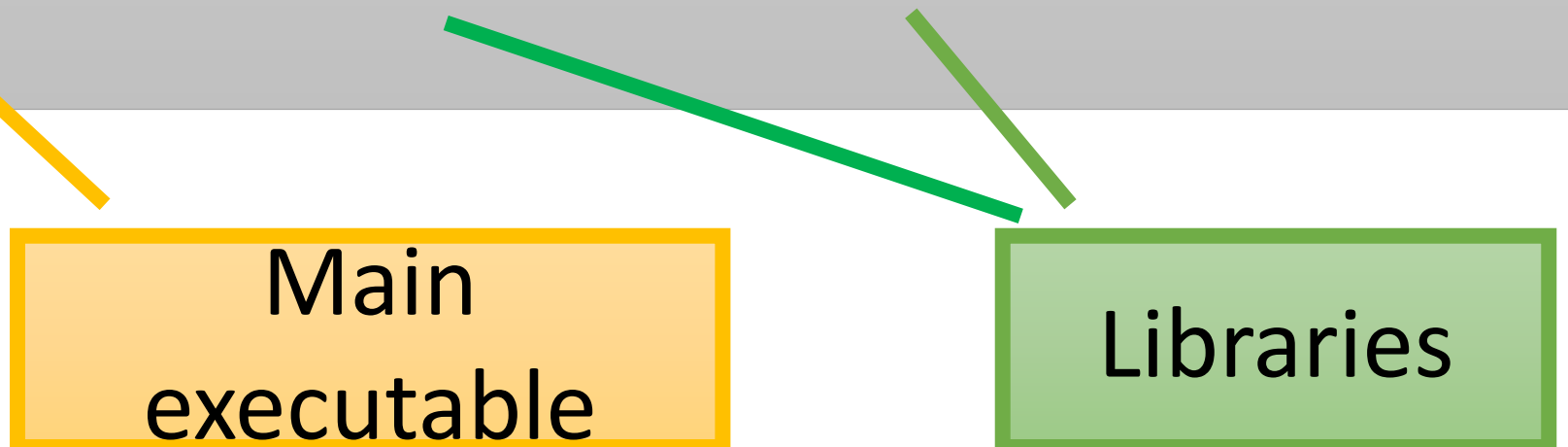
Installed in user defined directory
mydir/bin
mydir/lib & lib64

* Suitable for installation for a group

After installation

```
ls pyGenomeTracks-2.0

bin   lib   lib64
```



Shebang line would define which python to use:

```
#!/usr/bin/python2.7
```

```
#!/usr/bin/python3.6
```

```
ls pyGenomeTracks-2.0

bin  lib  lib64
```

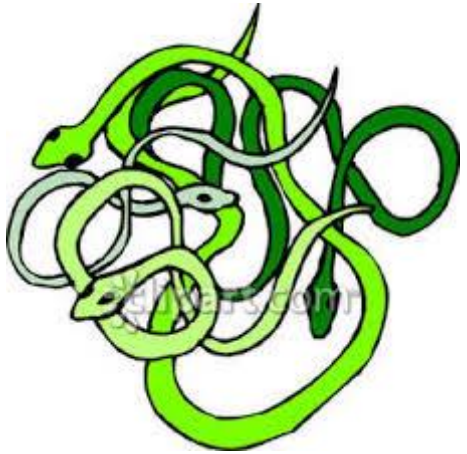
To run the software:

```
export PATH=/programs/pyGenomeTracks-2.0/bin:$PATH

export PYTHONPATH=/programs/pyGenomeTracks-
2.0/lib64/python2.7/site-packages:/programs/pyGenomeTracks-
2.0/lib/python2.7/site-packages/
```

PYTHON

After a while ….



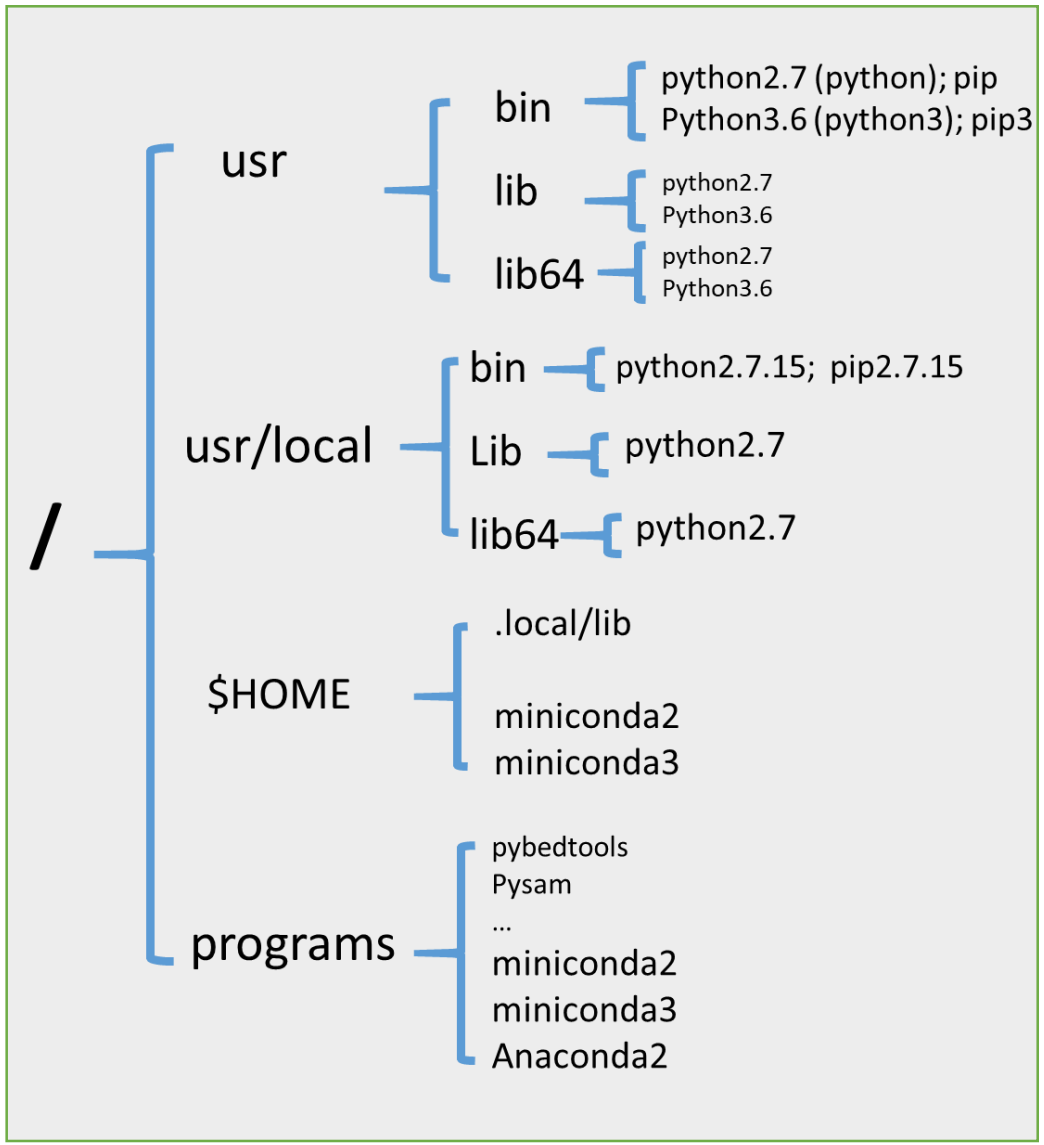
/

- usr
  - bin
    - python2.7 (python); pip
    - Python3.6 (python3); pip3
  - lib
    - python2.7
    - Python3.6
  - lib64
    - python2.7
    - Python3.6
- usr/local
  - bin
    - python2.7.15; pip2.7.15
  - Lib
    - python2.7
  - lib64
    - python2.7
- $HOME
  - .local/lib
  - miniconda2
  - miniconda3
- programs
  - pybedtools
  - Pysam
  - …
  - miniconda2
  - miniconda3
  - Anaconda2

Instructions for Python on BioHPC

https://biohpc.cornell.edu/lab/userguide.aspx?a=software&i=556#c

PYTHON



Precedence

Executables

$PATH

- /usr/local/bin
- /usr/bin

Libraries

$PYTHONPATH

- sys.path (installation-dependent default)

- `export PATH=/programs/pybedtools/bin:$PATH`
- `export PYTHONPATH=/programs/pybedbools/lib64`
- `unset PYTHONPATH`

Check which python module is being used

For example:

>>> import numpy

>>> print numpy.__file__
/usr/lib64/python2.7/site-packages/numpy/__init__.pyc

>>> print numpy.__version__
1.14.3

$PYTHONPATH frequently causes problem. E.g. python2 and python3 share the same $PYTHONPATH

* run these commands in “python” prompt

One common problem when running python:

PYTHONPATH

- Both PYTHON2 and PYTHON3 use the same variable PYTHONPATH;
- Some installation tool would automatically add PYTHONPATH setting in .bashrc

```
## set PYTHONPATH
export PYTHONPATH=/programs/pysam/lib/python2.7/site-packages

## check PYTHONPATH
echo $PYTHONPATH

##clear PYTHONPATH
unset PYTHONPATH
```

When installing software, PIP might up- or downgrade some modules as needed

The could cause problems, as modules are shared by many software.

e.g. when installing deepTools, **requirements.txt** in deepTools package would tell pip to install/upgrade following modules.

```
numpy>=1.9.0
scipy>=0.17.0
matplotlib>=2.1.2
pysam>=0.14.0
py2bit>=0.2.0
numpydoc>=0.5
pyBigWig>=0.2.1
plotly>=1.9.0
```

How PIP handle required modules?

Default behavior:

Missing modules -> Install

Existing modules

- **Version acceptable: skip**
- **Version not acceptable: uninstall and install right version.**

Existing modules

- Version acceptable: skip
- Version not acceptable: uninstall and install right version

- For root user, there is a risk of downgrade a module to an old version;
- For non-root user, you will get an error message of permission denied;

PIP parameters to change default behavior

--user

- Will not un-install existing modules;
- Skip modules meet requirement;
- Install required module that do not exist;

Parameters to change default behavior

--ignore-installed (-I)

- Install all required modules, present or not;
- Together with **--install-option="--prefix=mydir"**

PIP parameters to change default behavior

--upgrade : Upgrade package and all required modules to latest version

Uninstall python module

```
pip uninstall numpy
```

Or, simply delete the ~/.local/lib/ directory or sub-directories

Install python package not present in PYPI

-- follow instructions by author

- Download files;
- Run command:

python setup.py install --prefix=/home/qisun/myPython

- Set PYTHONPATH before running code;

PERL

	PYTHON	PERL	R
Repository	PYPI	**CPAN**	CRAN
Installation tool	pip	**cpan**	install.packages
Library PATH	PYTHONPATH	**PERL5LIB**	

PERL

Check whether a PERL module is present

```
perl -e "use Bio::Seq"
```

or

```
perldoc Bio::Seq
```

Install PERL modules from CPAN

https://cbsu.tc.cornell.edu/lab/doc/Install_PERL_modules.pdf

1. Configure cpan - specify the directory to install;

2. Use cpan to install PERL modules:

```
install XML::Simple
or
force install XML::Simple
```

Configuration of cpan

- Default cpan configuration is ok, and will install PERL modules into your home directory: $HOME/perl

- To reset cpan configuration, you can delete the whole cpan configuration directory: $HOME/.local/share/.cpan

Specify paths of PERL modules installed by you

export PERL5LIB=$HOME/perl/lib/perl5

PERL would search these paths to find a module:

1. **$PERL5LIB defined directories;**

2. **@INC defined directories;**
   Use this command to check:
   perl "-e print @INC“

R

	PYTHON	PERL	R
Repository	PYPI	CPAN	**CRAN**
Installation tool	pip	cpan	**install.packages**
Library PATH	PYTHONPATH	PERL5LIB	

R

Different versions of R on BioHPC

```
ls /programs |grep "^R-"
R-2.13.0
R-2.15.0
R-2.15.2
R-3.0.1
R-3.0.1a
R-3.0.2
R-3.1.0
R-3.2.2
R-3.2.2p
R-3.2.2s
R-3.2.5
R-3.2.5s
R-3.3.2
R-3.3.2s
R-3.4.1
R-3.4.1s
R-3.4.2
R-3.4.2s
```

R Command: R

Default: R-3.4.2

To use another version of R:
export PATH=/programs/R-3.4.1/bin:$PATH

- This is also applicable to using Rscript command to run R script.

On BioHPC, two separate installations for each R version

R-3.4.2 (Default): parallel BLAS library

R-3.4.2s: regular BLAS library

- Parallel BLAS (Basic Linear Algebra Subprograms) reduces computing time for linear algebra calls by a factor of 3 or more;
- Parallel BLAS could cause '**illegal operand**' errors for some packages;

Install R packages from CRAN

```
# install R package
install.packages("GD")

# load R package
library(GD)
```

* R packages are only installed to the specific version of R you are using.

Check version and path of R packages

Check version: **packageVersion("edgeR")**

Find package location: **find.package("edgeR")**

Check search path: **.libPaths()**

Install R packages from GitHub

```
llibrary(devtools)

install_github("rqtl/qtl2geno")
```

C / C++

Versions of the GCC compiler

Default gcc: 4.8.5

Other versions: 5.5.0, 7.3.0 (/usr/local/gcc-*)

To use a different version of GCC

```
export PATH=/usr/local/gcc-7.3.0/bin:$PATH

export LD_LIBRARY_PATH=/usr/local/gcc-7.3.0/lib:/usr/local/gcc-7.3.0/lib64
```

General installation procedure

1. **Configure** - customize/verify compilation instruction;

2. **Compile** - from source code to binary code;

3. **Install** - put the executable/library in right locations;

Pre-built Binary

vs

Compilation From Source Code

Why compilation?

1. Pre-built binary code not provided;

2. Compilation would optimize CPU/GPU usage
(e.g. software developed for GPU)

* Sofware package manager, e.g. conda cannot do compilation, only install pre-built code;

Configure – Part 1

```
./configure
```

1. Specify installation directory;

To change: ./configure --prefix=/home/xxxxx/bin

2. Verify the compiler, libraries;

Deal with it if libraries missing or not in right version
(e.g. set LD_LIBRARY_PATH)

Configure – Part 2

Another system for configuration:

cmake

or

cmake3

Most software requires cmake v3. The command on BioHPC is cmake3

Specify installation directory

```
cmake3 -DCMAKE_INSTALL_PREFIX:PATH=/home/xxxx/bin
```

Configure – Part 3

Manually edit the makefile

Install

```
make install
```

To run the software:

- Modiy $PATH or use full path of the executable;
- Modify $LD_LIBRARY_PATH if custom libraries installed;

Environment variable for executables path

$PATH

Environment variables for custom library path

PYTHON	**$PYTHONPATH**
PERL	**$PERL5LIB**
C	**$LD_LIBRARY_PATH**

Set environment variable:
export PATH=mydirectory:**$**PATH

Check environment variable:
echo **$**PATH