How to Install Software on Abel (and Colossus)

Bjørn-Helge Mevik

Research Infrastructure Services Group, USIT, UiO

RIS Course Week

General Points

- We encourage people to install their software themselves, especially `R`, Perl and Python packages.
- Make sure you have loaded the modules you need, all the modules you need, and only the modules you need:

```
$ module purge
$ module load foo bar
$ module list
```

- Try to use the Intel compilers when compiling. That is the default when installing `R` and Python packages:

```
$ module load intel
$ module list
```

R Packages

- Main repository: `https://cran.r-project.org/`
- Our documentation: `https://www.uio.no/english/services/it/research/hpc/abel/help/software/R.html`
- Tip: Avoiding Selecting Repository Every Time: Add

```
local({
    r <- getOption("repos")
    ## For R < 3.2.3:
    #r["CRAN"] <- "http://cran.uib.no"
    ## For R >= 3.2.3:
    r["CRAN"] <- "https://cran.uib.no"
    options(repos = r)
})
```

to your `~/.Rprofile` file.

CRAN Packages

https://cran.r-project.org/

```
$ module purge
$ module load R/3.3.2
$ module list
Currently Loaded Modulefiles:
  1) intel/2017.0   3) zlib/1.2.8     5) xz/5.2.2       7) R/3.3.2
  2) curl/7.46.0    4) bzip2/1.0.6    6) pcre/8.38
$ R
[...]
> install.packages("pls")
[...]
* DONE (pls)

The downloaded source packages are in
‘/tmp/RtmpydUdGZ/downloaded_packages’
>
```

See `?install.packages` for details.

Packages with Dependencies

Some packages depend on libraries etc. outside R.

```
> install.packages("rjags")
[...]
checking for pkg-config... /usr/bin/pkg-config
configure: WARNING: pkg-config file for jags 4 unavailable
configure: WARNING: Consider adding the directory containing ‘jags.p
configure: WARNING: to the PKG_CONFIG_PATH environment variable
configure: Attempting legacy configuration of rjags
checking for jags... no
configure: error: "automatic detection of JAGS failed. Please supply
ERROR: configuration failed for package ‘rjags’
* removing ‘/cluster/home/bhm/nobackup/R/x86_64-pc-linux-gnu-library

The downloaded source packages are in
‘/tmp/RtmpFyVuNK/downloaded_packages’
Warning message:
In install.packages("rjags") :
  installation of package ‘rjags’ had non-zero exit status
```

Packages with Dependencies (II)

```
$ module avail jags
[...]
jags/3.3.0                      jags/4.2.0
jags/4.0.0                      jags/4.2.0_2015_3(default)
$ module load jags/4.2.0_2015_3
$ module list
$ R
[...]
> install.packages("rjags")
[...]
* DONE (rjags)
[...]
>
```

Remember to load the modules before using the package later!

Other Packages

- Bioconductor packages (http://bioconductor.org/install/)

```
> source("http://bioconductor.org/biocLite.R")
> biocLite(c("AnnotationDbi"))
```

- Local files

```
$ wget https://cran.r-project.org/src/contrib/abind_1.4-5.tar.g
$ R CMD INSTALL abind_1.4-5.tar.gz
```

See `R CMD INSTALL --help` for details.

Advanced Options

- Install to a different location:
  `R CMD INSTALL --library=/the/path` or
  `install.packages(..., lib = "/the/path")`.
  `.libPaths("/the/path")` to use location.

- Some packages will not compile with Intel. Then use an R built with Gnu compilers:

```
$ module purge
$ module load R/3.3.2.gnu
$ module list
```

  Often, the package can later be *used* with the standard R.

- Colossus (on your TSD LinuxVM):

```
> install.packages("package",
                   repos = "file://tsd/shared/R/cran/")
```

Perl Packages

- Main repository: `http://www.cpan.org/`
- Our documentation: `https://www.uio.no/english/services/it/research/hpc/abel/help/software/Perl_modules.html`
- Setting up directories and environment variables:

```
$ module load perlmodules  # local::lib and cpanm
$ eval $(perl -Mlocal::lib)
$ export PERL_CPANM_HOME=/tmp/cpanm_$USER
```

- The above can be added to `~/.bashrc` or `~/.bash_profile`.

CPAN Packages

```
$ cpanm JSON
--> Working on JSON
Fetching http://www.cpan.org/authors/id/M/MA/MAKAMAKA/JSON-2.90
Configuring JSON-2.90 ... OK
Building and testing JSON-2.90 ... OK
Successfully installed JSON-2.90
1 distribution installed
$ perl -MJSON -e 'print "ok\n"'
ok
```

(Try Align::Sequence)

Advanced Options

- Install from local file:
  ```
  $ wget \
  http://www.cpan.org/authors/id/M/MA/MAKAMAKA/JSON-2.90.tar.gz
  $ cpanm JSON-2.90.tar.gz
  ```
- Install to a separate directory:
  ```
  eval $(perl -Mlocal::lib=/the/path)
  cpanm Some::Module
  ```
  (Good for keeping projects separate.)
- It is possible to avoid `module load perlmodules` every time. See `https://www.uio.no/english/services/it/research/hpc/abel/help/software/Perl_modules.html`
- Colossus: Currently no CPAN mirror inside TSD, so packes must be imported and installed from local file.

Python Packages

- Main repository: `https://pypi.python.org/`
- Our documentation:
  - Python 2: `https://www.uio.no/english/services/it/research/hpc/abel/help/software/Python%202.html`
  - Python 3: `https://www.uio.no/english/services/it/research/hpc/abel/help/software/Python%203.html`

Python 2

```
$ module purge
$ module load python2/2.7.10
$ module list
Currently Loaded Modulefiles:
  1) intel/2015.3     2) libffi/3.0.13     3) python2/2.7.10
$ pip install --user dexml
Collecting dexml
  Downloading dexml-0.5.1.tar.gz
Installing collected packages: dexml
  Running setup.py install for dexml
Successfully installed dexml
$
```

See `pip --help` and `pip install --help` for details.

Python 3

```
$ module purge
$ module load python3/3.5.0
$ module list
Currently Loaded Modulefiles:
  1) intel/2016.0    2) libffi/3.0.13    3) python3/3.5.0
$ pip3 install --user dexml
Collecting dexml
  Using cached dexml-0.5.1.tar.gz
Installing collected packages: dexml
  Running setup.py install for dexml
Successfully installed dexml
$
```

See `pip3 --help` and `pip3 install --help` for details.

Packages with Dependencies

```
$ module purge
$ module load python2/2.7.10
$ module list
$ pip install --user mpi4py
[...]
    error: Cannot compile MPI programs. Check your configuratio
[...]
$ module load openmpi.intel/1.10.2
$ pip install --user mpi4py
[...]
Successfully installed mpi4py-2.0.0
```

Remember to load the modules before using the package later!

Advanced Options

- Install from local file:
  ```
  wget \
  https://pypi.python.org/packages/source/d/dexml/dexml-0.5.1.tar
  pip install --user dexml-0.5.1.tar.gz
  ```
- Install in separate directory (good for separating projects):
  ```
  pip install --target=/the/path thepackage
  PYTHONPATH=/the/path:$PYTHONPATH
  ```
- Colossus (on your TSD Linux VM):
  ```
  pip install --user \
    --index-url=file:///shared/pypi/mirror/web/simple package
  ```

(Or `pip3`.)

General Software Packages

- You can install software in your home directory, or in a shared project area you have access to.
- We recommend installing from source if possible.
- If a package depends on some software or library that is installed on Abel, use `module load` before installing it, and before using it later.

Different Types of Installs

- How to install depends on what the developers have decided. *Read the documentation!* There is no alternative.
- Some common types of installs:
  - GNU Autoconf based
  - Cmake based
  - Manually written Makefile
  - Binary install in Tar og Zip file
- Note: Ubuntu/Debian binary packages cannot be installed, since Abel and Colossus run RedHat (CentOS). (So *no* `apt-get` or `aptitude`.)
- It might be possible to install RPMs, but it is tricky

GNU Autoconf Based Install

```
$ module purge
$ module load intel/2017.1
$ module list
$ wget \
ftp://heasarc.gsfc.nasa.gov/software/fitsio/c/cfitsio3370.tar.gz
$ tar xf cfitsio3370.tar.gz
$ cd cfitsio/
$ less README
$ ./configure --prefix=$HOME/lib/cfitsio
$ make
$ make install
```

- Some packages have a `make check` to test the build
- Some packages can be run from the build area without using `make install`
- See `./configure --help` for more options

Cmake

- With `cmake`, one can choose installation directory with `cmake -DCMAKE_INSTALL_PREFIX=/the/path`.
- Example: `bamm`

```
$ wget https://github.com/macroevolution/bamm/archive/v2.5.0.ta
$ tar xf v2.5.0.tar.gz
$ cd bamm-2.5.0; ls
$ less README.md
$ module purge
$ module load intel/2017.1 cmake/3.1.0
$ module list
$ mkdir build; cd build
$ cmake -DCMAKE_INSTALL_PREFIX=$HOME/bamm ..
$ make -j
[...]
/usr/include/c++/4.4.7/c++0x_warning.h(31): error: #error direc
This file requires compiler and library support for the upcomin
ISO C++ standard, C++0x. [...]
```

Cmake (II): bamm, take 2

- Let's try with gcc instead:

```
$ cd ..
$ rm -rf build
$ module purge
$ module load gcc/6.1.0 cmake/3.1.0
$ module list
$ mkdir build; cd build
$ cmake -DCMAKE_INSTALL_PREFIX=$HOME/bamm ..
$ make -j
[...]
[100%] Built target bamm
$ make install
[...]
-- Installing: /usit/abel/u1/bhm/bamm/bin/bamm
$ ~/bamm/bin/bamm
Usage: bamm -c <control-file> [--<parameter-name> <parameter-va
```

Manual Makefile

- The documentation (`README`, `INSTALL`, etc.) should give you the information needed.
- Well written Makefiles use a variable to control the installation directory. Then one can use `make` *`VARIABLE`*`=/the/path install` to build and install.
- If the Makefile uses an environment variable, one can set it with `export` *`VARIABLE`*`=/the/path` before running `make`.
- If the installation directory is hard coded in the Makefile, one must edit the file.

Manual Makefile (II): subread

- Example: subread

```
$ wget http://downloads.sourceforge.net/project/subread/\
   subread-1.5.0-p1/subread-1.5.0-p1-source.tar.gz
$ tar xf subread-1.5.0-p1-source.tar.gz
$ cd subread-1.5.0-p1-source/; ls; less README.txt
$ cd src; less Makefile.Linux
$ module purge
$ module load gcc/6.1.0; module list
$ make -f Makefile.Linux
[...]
#  Installation finished.                                    #
#  Generated executables were copied to directory ../bin/ #
[...]
$ # optional: mv ../bin ~/<somewhere>
$ ../bin/exactSNP
Version 1.5.0-p1
Usage:
./exactSNP [options] -i input -g reference_genome -o output
[...]
```

Binary Installs

- Usually Tar or Zip files.
- Often, one only needs to unpack the files. Optionally, one can rename the directory, or copy the extracted files to the desired directory.
- Tip: First check whether the file unpacks in a subdirectory or into the current directory (`tar tvf file.tar.gz` or `unzip -t file.zip`).
- There is a risk that the programs need libraries that are missing, too old or too new on cluster. Then one needs to install these as well, or switch to installing from source, if possible.