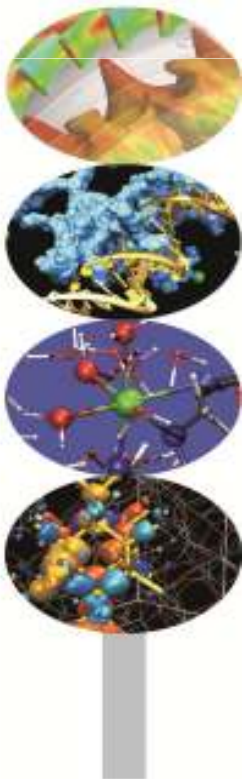
CINECA SCAI
SuperComputing Applications and Innovation

Software modules and programming environment

SCAI User Support



Marconi User Guide
https://wiki.u-gov.it/confluence/display/SCAIUS/UG3.1%3A+MARCONI+UserGuide

CINECA

Modules

Set module variables

- module load <module_name>

Show module variables, prereq (compiler and libraries) and conflict

- module show <module_name>

Give info about the software, the batch script, the available executable variants (e.g, single precision, double precision)

- module help <module_name>

Modules

Set the variables of the module and its dependencies (compiler and libraries)

- module load autoload <module name>

module load <compiler_name>

module load <library _name>

module load <module_name>

Profiles and categories

- The modules are
  - organized by functional category (compilers, libraries, tools, applications,…).
  - collected in different profiles (base, advanced, …..)
- The profile is a group of modules that share something

Profiles

- Programming profiles
  - They contain only **compilers, libraries and tools** modules used for compilation, debugging, profiling and pre-proccessing activity

BASE

Basic compilers, libraries and tools (gnu, intel, intelmpi, openmpi--gnu, math libraries, profiling and debugging tools, rcm,…)

ADVANCED

Advanced compilers, libraries and tools (pgi, openmpi--intel, ….)

Profiles

- Production profiles
  - They contain only **libraries, tools and applications** modules used for production activity
  - They match the most important scientific domains. For this reason we call them “domain” profiles:

CHEM
PHYS
LIFESC
BIOINF
ENG

Profiles

Programming and production and profile

- ARCHIVE
  - It contains all old versions of applications and tools and the old versions of compilers and libraries not used by other modules

Profiles

- BASE profile is the default. It is automatically loaded after login:

```
module av
```

To use a module of BASE profile you have only to load the module:

```
module load autoload <module_name>
```

To use a module placed under others profiles, you have to load preventively the corresponding profile:

```
module load profile/<profile_name>
module load autoload <module_name>
```

To list all loaded profiles use the following command:

```
module list
```

Mappa dei moduli

- modmap command
  - -all
    Print the full modules map: all profiles, all categories defined for each profile, all modules available for each category
  - -profiles
    Print all profiles available
  - -categories
    Print all categories available
  - -p <profile name>
    Print all modules available for the selected profile
  - -c <category name>
    Print all modules available for the selected category
  - -m <module name>
    Print all versions of the selected module

Mappa dei moduli

- NAMD - use example:

Find out all available NAMD versions and the profiles where they are located

```
modmap -m namd
```

Load the profile where NAMD of my personal interest is located

```
module load profile/chem    or
module load profile/archive
```

Load NAMD module

```
module load autoload namd
```

Load a specific version if more NAMD versions are available

```
module load autoload namd/<version>
```

Compilers

modmap -c compilers

Profiles: base

gnu

5.3
6.1

intel

pe-xe-2016--binary—binary

intelmpi

5.1

openmpi

1.10--gnu—6.1

python

3.5.1

Profiles: advanced

pgi

16.3

For each compiler more versions can be available into the specific profile. The default module is the latest version

INTEL

- List of compilers:

ifort: Fortran77 and Fortran90 compiler
icc: C compiler
cpc: C++ compiler

- Example of usage:

module load intel

icc -o my_exe prog.c
ifort -o my_exe prog.f90

GNU

- List of compilers:

gfortran: GNU Fortran compiler
gcc: GNU C compiler
g++: GNU C++ compiler

- Example of usage:

```
module load gnu
gcc -o a.out foo.c
```

IntelMpi

- List of wrappers using INTEL compilers:

mpiifort (Fortran MPI wrapper)
mpiicc (C MPI wrapper)
mpiicpc (C++ MPI wrapper)

- List of wrappers using GNU compilers:

mpif77,mpif90: (Fortran MPI wrapper)
mpicc: (C MPI wrapper)
mpicxx: (C++ MPI wrapper)

- Example of usage:

```
module load autoload intelmpi
mpiicc -o my_mpi_exe mpi_prog.c
```

Openmpi--gnu

- List of wrappers:

mpif77,mpif90: (Fortran MPI wrapper)
mpicc: (C MPI wrapper)
mpicxx: (C++ MPI wrapper)

- Example of usage:

```
module load autoload openmpi
mpicc -o my_mpi_exe mpi_prog.c
```

Python and its packages

PYTHON

- module load python
  - python

- PACKAGES
- 1. module load python
  - /…/site-packages → they contain only basic packages

- 2. module load <module_name>

- module load numpy
- module load scipy
- module load ipython
- module load cython

Python and its packages

- 3. VIRTUALENV to create isolated *Python* environments. Follow these steps:

using python interpreter from the module

- module load python

create a virtualenv, basically just a new directory (my_venv) containing all you need

- virtualenv my_venv

activate the new virtualenv

- source my_venv/bin/activate

now you can pip install whatever you need

- pip install **matplotlib**

when you are done working

- deactivate

Python and its packages

if you want to dump all the pip installed modules on your virtualenv

- pip freeze > requirements.txt

if you want to recreate tha same virtualenv

- pip install -r requirements.txt

if you want to remove the virtualenv

- rm -rf my_venv

Libraries

modmap -c libraries

Profiles: base

- blas
- boost
- fftw
- hdf5
- lapack
- mkl
- mpi4py
- netcdf
- numpy
- scalapack
- scipy
- szip
- zlib

Libraries

- The available libraries are compiled with **intel compiler** when they need a good performance, otherwise they are compiled with **gnu compiler** in order to ensure the the compatibility gnu/intel

- For each library more versions can be available into the specific profile. The default module is the latest and parallel version

- HDF5 - USE EXAMPLE

```
module load autoload hdf5
module load autoload szip
module load autoload zlib
```

$ icc **-I$HDF5_INC** input.c **-L$HDF5_LIB -lhdf5 -L$SZIP_LIB -lsz -L$ZLIB_LIB -lz**

Tools

modmap -c tools

Profiles: base

- cmake
- rcm
- scalasca
- superc
- totalview
- valgrind
- vtune

For each tool only one module is available into the specific profile. It is the latest version. The older versions are available into archive profile

CINECA SCAI
SuperComputing Applications and Innovation

Applications

modmap -c applications

Profiles: eng

abaqus
ansys
openfoam
starccm++

Profile: chem

abinit
adf
amber
cp2k
cpmd
crystal
dl_poly
g09
gromacs
lammps
molcas
namd
nwchem
plumed
pyfrag
qe
siesta
vasp
yambo

Profile: bioinf

r

CINECA

Applications

- For each application only one module is available into the specific domain. It is the latest version. The older versions are available into archive profile

- For the same application more variants can be available and they differ for the executable name into the corresponding module

Gromacs EXAMPLE:

module load gromacs

→ executables:

mdrun
mdrun_mpi
mdrun_mpi_plumed