Accessing Software

- Policies
- Installed software
- Modules
- Installing additional software
- Using and Installing Perl
- Using and Installing Python
- Using R Packages
- Using Matlab
- Compilers

Policies

The policies regarding the installation of software is on this page. In general, scientific software is installed as requested with the caveats noted in that section.

Installed software

A list of installed software is kept at this page.

On Ocelote, many libraries like FFTW and MPICH came with the Cluster Manager. As a consequence, they have different naming from the other clusters. For example, `module load blas` on the older clusters will need to be `module load blas/gcc/64`. There are frequently versions of these libraries that have several compilation options, for example, blas was created with four different compilers on Ocelote.

Modules

Many popular software packages are installed and available as modules. There may be several versions of a package available.

<table>
<thead>
<tr>
<th>Module Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>module avail</td>
<td>Display all the software and versions installed on the system</td>
</tr>
<tr>
<td>module list</td>
<td>Display the software you have loaded in your environment</td>
</tr>
<tr>
<td>module load modulenname</td>
<td>Load a software module in your environment</td>
</tr>
<tr>
<td>module purge</td>
<td>Unload all the software modules from your environment</td>
</tr>
<tr>
<td>module unload modulenname</td>
<td>Unload a specific software package from your environment</td>
</tr>
<tr>
<td>module help</td>
<td>Display a help menu for the module command</td>
</tr>
</tbody>
</table>

Installing additional software
To submit a request to have software installed on the UA HPC systems use the HPC Software Install Request form: [http://uits.arizona.edu/forms/hpc-software-install-request](http://uits.arizona.edu/forms/hpc-software-install-request)

You can install software packages into your home directories with the space that is allocated to you with your HPC account. However, you cannot install software that requires root permission, or use a method like "yum install" that accesses system paths.

Follow this link for detailed information on how to install your own software.

**Using and Installing Perl**

Follow this link for more information on using Perl.

**Using and Installing Python**

Follow this link for more information on using Python.

**Using R Packages**

You can install your own R packages which is similar to using virtualenv with Python.

1. Make directory to store packages
   
   ```shell
   $ mkdir -p ~/R/library
   ```
2. Tell R where the directory is by creating an environment file:
   
   ```shell
   $ echo 'R_LIBS=~/R/library/' >> ~/.Renviron
   ```
3. For example to install and load the package "ggplot2":
   
   ```shell
   $ module load R
   
   $ R
   ...
   > install.packages("ggplot2")
   > library(ggplot2)
   ```
4. After this you’ll only need the library command to load your custom package
5. For more information:
   

**Using Matlab**

MATLAB performs its own hardware discovery and it might try to access all the cores and the memory of the node even if the full node wasn't allocated. That will result in scheduler killing the job. To prevent that the full Ocelote node of 28 cores and 168GB of memory should be allocated to run a MATLAB job.

Like any other application, MATLAB has to be loaded as a module before you can use it. To see all the installed versions of the MATLAB use command `module avail matlab`.

The typical procedure for performing calculations on UA HPC systems is to run your program non-interactively on compute nodes. The easiest way to run MATLAB non-interactively is to use input/output redirection. This method uses Linux operators `<` and `>` to point MATLAB to the input file and tell where to write the output (see the example script). The other method is to invoke MATLAB from the PBS script and execute specified statement using `-r` option. For details please refer to the manual page of `matlab` command:

#!/bin/bash
#PBS -N job_name
#PBS -W group_list=group_name
#PBS -q standard
#PBS -l select=1:ncpus=28:mem=168gb:pcmem=6gb
#PBS -l walltime=01:00:00
#PBS -l cput=28:00:00

cd $PBS_O_WORKDIR

module load matlab

matlab -nodisplay -nosplash < script_name.m > output.txt

The options -nodisplay and -nosplash in the example prevent MATLAB from opening elements of GUI. To view the full list of options for matlab command load the MATLAB module and type `matlab -h` in Linux prompt, or use the link above to the manual page on MathWorks website.

Compilers

- **ICE**
  There are several compilers available for your use. Remember when you run your code and you need to do a module load, that you use the same version of compiler that the code was originally compiled with.
  - GCC is available by default. `gcc --version` shows that it is 4.4.4. That version was installed at cluster installation and has not been updated to maintain consistency between compiling software and subsequently running it. If you also need the GNU Scientific Libraries (gsl), that is available using module load gsl which will get you version 1.15
  - The Intel 2012 compiler suite is available as a module. `module load intel` will load the 2012 suite. It is the default for the same reason as gcc. The math kernel libraries (mkl) are provided when you module load intel (any version) - no separate step is required.
  - The Intel 2013 compiler suite is also available as an optional module. `module load intel/2013.5.192` will provide that version if needed.
  - The latest Intel suite is now available with `module load intel/xe.2016.u2`. Unlike earlier versions, you do not need to separately load the MPI capable compiler. For detailed information, refer to the Intel documentation: https://software.intel.com/en-us/articles/intel-parallel-studio-xe-2016-release-notes

- **MPI compilers**
  - intel-mpi/2012.0.032 is the default version which matches the non-mpi version. Use `module load intel-mpi`
  - intel-mpi/2013.5.192 is available by specifying `module load intel-mpi/2013.5.192`
  - openmpi version 1.4.4 is available by specifying `module load openmpi`
  - mpich2 is available at version 1.4.1p1 by default or you can get version 3.1.4 with `module load mpich2/3.1.4`

- **El Gato**
  - The principles are similar for ElGato except that the intel and intel-mpi compilers are only available for the 2013 versions.
  - openmpi is available both for version 1.6.5 and version 1.8.1
  - El Gato has a separate web site with easy to follow instructions.

- **Ocelote**
- GCC is available without loading a module. `gcc --version` shows that it is 5.2.0. If you also need the GNU Scientific Libraries (gsl), that is available using `module load gsl` which will get you version 2.1.

- The Intel Compiler suite 2016 is available in both 32 and 64 bit versions. The math kernel libraries (mkl) are provided as separate modules, also in 32 and 64 bit versions.

- MPI compilers. There are more choices now so pay attention.
  - There are standard Red Hat versions of mpich, mvapich, mvapich2 and openmpi. Some extra options are invoked when you load one - use `module avail` to see the specific name.
  - The same four compilers are available with more detailed options for gcc, intel and open64. Again use `module avail` for the appropriate choices.

- AVX2 The new cluster has Intel V3 Haswell processors. A key feature of these is AVX2. Read this Intel document.

  For AVX2 support, compile with the `-xHOST` option. Note that `-xHOST` alone does not enable aggressive optimization, so compilation with `-O3` is also suggested. The `-fast` flag invokes `-xHOST`, but should be avoided since it also turns on interprocedural optimization (`-ipo`), which may cause problems in some instances.

  For GNU compilers, AVX support is only available in version 4.6 or later. For AVX support, compile with `-mavx`