Installing LMGC90 on Cedar and Graham Compute Canada servers

November 7, 2020

Disclaimers

If you need to learn how to access ComputeCanada, how servers are structured or how to upload and download data on clusters, we recommend that you refer to our original instruction manual or the inexhaustible source of information ComputeCanada_wiki.

If you plan to use Béluga or Niagara, we also recommend that you refer to the previous notice which will give you exhaustive instructions to install LMGC90 and start learning its use on clusters.

If you plan to use Cedar or Graham, follow the instructions below.
NOTE1: If you never worked on ComputeCanada servers with LMGC90, we advise you to begin with small steps on Béluga.
NOTE2: Use one of the version of LMGC90 available on the website.

ComputeCanada is the national network of servers for high-performance computing. It is composed of 4 main servers:

- Cedar
- Graham
- Béluga
- Niagara

Each one of them is provided a set of modules which allow you to switch between different versions of software packages you may want to use. This root set is called a "standard environment" and is named `StdEnv`. At the present time it exists three competing versions of these `2016.4`, `2018.3` and `2020`. For those of you who might want to investigate what truly lies behind, learn that detailed information on the major additions developed between each version can be find here.

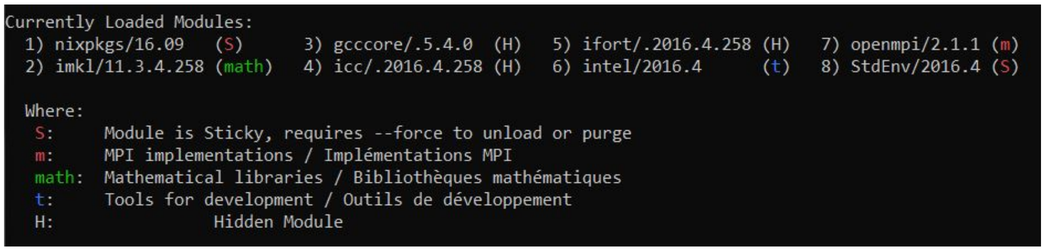
`StdEnv/2016.4` is the standard environment originally implemented on both Cedar and Graham. It explains why the implementations of LMGC90 on those clusters needs a different handling procedure.

Modules in the server

First and foremost, we remind you that at any time and in any repertory of your server share you can access to the list of modules loaded by using the command:

```
[name@server ~]$ module list
```

By doing so you must obtain an initial structure similar to this one:



```
Currently Loaded Modules:
  1) nixpkgs/16.09   (S)      3) gcccore/.5.4.0  (H)   5) ifort/.2016.4.258 (H)   7) openmpi/2.1.1 (m)
  2) imkl/11.3.4.258 (math)   4) icc/.2016.4.258 (H)   6) intel/2016.4      (t)   8) StdEnv/2016.4 (S)

  Where:
   S:     Module is Sticky, requires --force to unload or purge
   m:     MPI implementations / Implémentations MPI
   math:  Mathematical libraries / Bibliothèques mathématiques
   t:     Tools for development / Outils de développement
   H:                Hidden Module
```

NOTE: If it is not the case, do not worry and stick to the following instructions.

At this point, you need to have uploaded LMGC90 data in a definite folder in such location:

```
[name@server ~]$ cd projects/account_name/name/lmgc90_
folder/
```

NOTE: If you need to get your location, don't forget to use the command:

```
[name@server ~]$ pwd
/the/corresponding/path/of/your/location
```

To load the modules that allow us to compile LMGC90, write the following line:

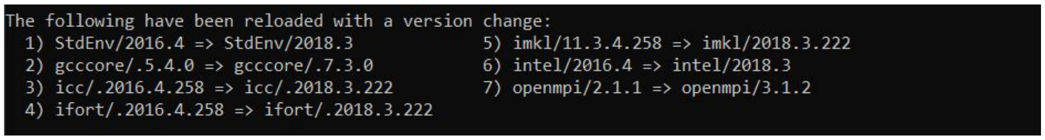
```
[name@server lmgc90_folder]$ module purge A
```

Now, understand that the compilation of LMGC90 requires specific modules but that the root set `StdEnv/2016.4` is unable to provide the proper versions of such material. In rare circumstances it may be then necessary to load a different standard environment to access particular versions of needed packages. We are actually looking for its next update named `StdEnv/2018.3` which you can load by overwriting the former environment:

```
[name@server lmgc90_folder]$ module load StdEnv/2018.3 B
```

NOTE: If the one above does not work, you may try:

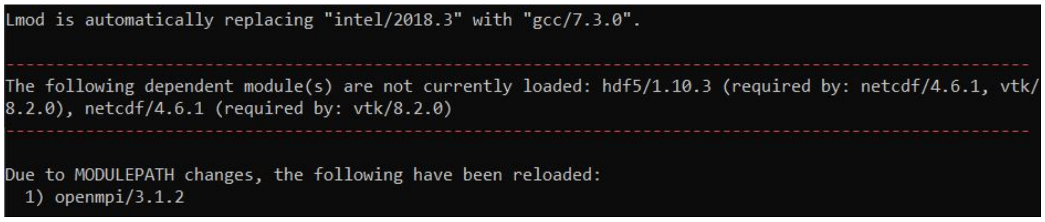
```
[name@server lmgc90_folder]$ module switch StdEnv/2016.4
StdEnv/2018.3 B*
```



```
The following have been reloaded with a version change:
  1) StdEnv/2016.4 => StdEnv/2018.3              5) imkl/11.3.4.258 => imkl/2018.3.222
  2) gcccore/.5.4.0 => gcccore/.7.3.0            6) intel/2016.4 => intel/2018.3
  3) icc/.2016.4.258 => icc/.2018.3.222          7) openmpi/2.1.1 => openmpi/3.1.2
  4) ifort/.2016.4.258 => ifort/.2018.3.222
```

Also, add:

```
[name@server lmgc90_folder]$ module load vtk nixpkgs/16.09
gcc/7.3.0 hdf5 C
```

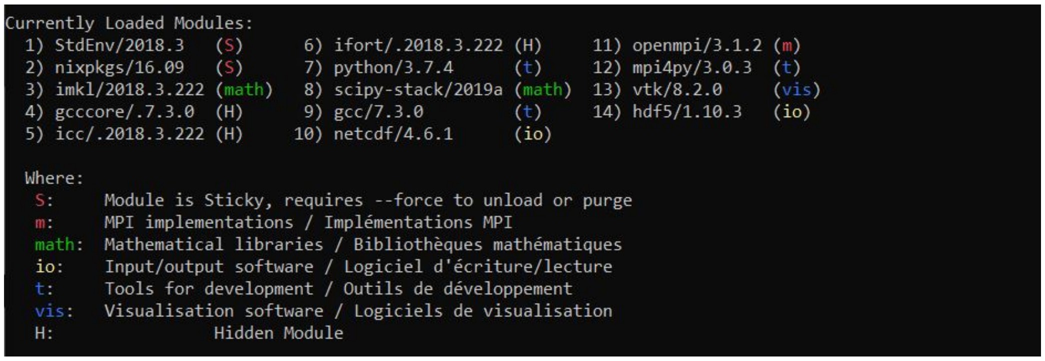


```
Lmod is automatically replacing "intel/2018.3" with "gcc/7.3.0".

-------------------------------------------------------------------------------------------------------
The following dependent module(s) are not currently loaded: hdf5/1.10.3 (required by: netcdf/4.6.1, vtk/
8.2.0), netcdf/4.6.1 (required by: vtk/8.2.0)
-------------------------------------------------------------------------------------------------------

Due to MODULEPATH changes, the following have been reloaded:
  1) openmpi/3.1.2
```

Hopefully, you have obtained this type of response.



```
Currently Loaded Modules:
  1) StdEnv/2018.3    (S)      6) ifort/.2018.3.222 (H)     11) openmpi/3.1.2 (m)
  2) nixpkgs/16.09    (S)      7) python/3.7.4      (t)     12) mpi4py/3.0.3  (t)
  3) imkl/2018.3.222 (math)    8) scipy-stack/2019a (math)  13) vtk/8.2.0     (vis)
  4) gcccore/.7.3.0  (H)       9) gcc/7.3.0         (t)     14) hdf5/1.10.3   (io)
  5) icc/.2018.3.222 (H)      10) netcdf/4.6.1      (io)

  Where:
   S:     Module is Sticky, requires --force to unload or purge
   m:     MPI implementations / Implémentations MPI
   math:  Mathematical libraries / Bibliothèques mathématiques
   io:    Input/output software / Logiciel d'écriture/lecture
   t:     Tools for development / Outils de développement
   vis:   Visualisation software / Logiciels de visualisation
   H:                Hidden Module
```

You will need to run the same three lines (A, B or B* and C) before running your simulations. In fact, you will systematically add them in each bash script from which we describe the structure later.

Installing LMGC90

Adequate modules are now loaded. The compilation is done as explained below following the procedure given by the LMGC90_wiki page.

Create a build folder with:

```
[name@server lmgc90_folder]$ mkdir build
```

Access to the build:

```
[name@server lmgc90_folder]$ cd build
```

And compile with:

```
[name@server build]$ cmake ..  -DMATLIB_VERSION=none
-DSPARSE_LIBRARY=none
```

Finally:

```
[name@server build]$ make
```

The compilation went probably well, without raising any issue to your attention.

Running your simulations with the queuing assistant: SLURM

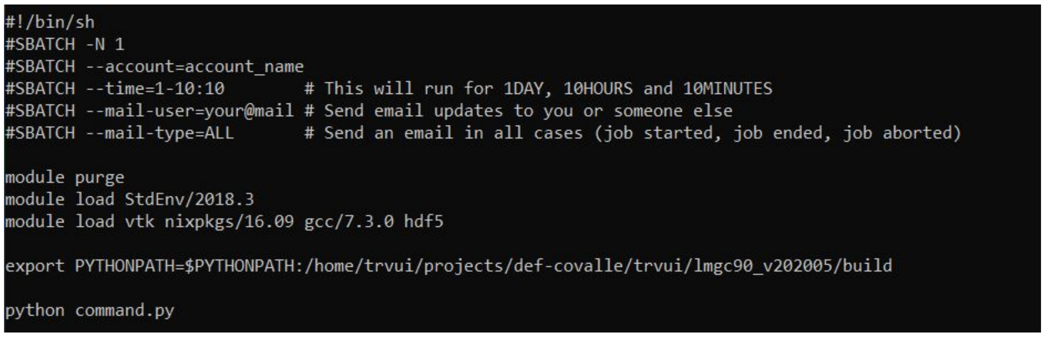
You must use a queuing assistant called SLURM to run your simulations from one of the two clusters we are interested in. We propose to work with a bash script. This same script will have to be in the folder of your simulation located somewhere in your `scratch`:

```
[name@server ~]$ cd scratch/path/simu_folder/
[name@server simu_folder]$ pwd
/home/name/scratch/path/simu_folder
```

After you have uploaded the `DATBOX` folder and the `command.py` referring to your simulation in its folder, create the bash script with the command:

```
[name@server simu_folder]$ nano script.sh
```

In the nano editor you are then able to write a simple bash script from which we give you an example of structure below.



```
#!/bin/sh
#SBATCH -N 1
#SBATCH --account=account_name
#SBATCH --time=1-10:10        # This will run for 1DAY, 10HOURS and 10MINUTES
#SBATCH --mail-user=your@mail # Send email updates to you or someone else
#SBATCH --mail-type=ALL       # Send an email in all cases (job started, job ended, job aborted)

module purge
module load StdEnv/2018.3
module load vtk nixpkgs/16.09 gcc/7.3.0 hdf5

export PYTHONPATH=$PYTHONPATH:/home/trvui/projects/def-covalle/trvui/lmgc90_v202005/build

python command.py
```

- The first line says that this is a bash script.
- Second line states that this belongs to the group `account_name`.
- Third line states the duration of the simulation. In Cedar and Graham the limit is set to 28 days.
- Fourth and fifth lines are optional but helpful. They will keep you updated of your simulation's state by email so you do not have to keep an eye on it constantly.
- Then, the modules are purged and loaded as required.
- Next line states where the sources of LMGC90 are located.
- And finally, we run the simulation.

Then, run it using:

```
[name@server simu_folder]$ sbatch script.sh
```

NOTE: If you want to learn more about the SLURM options, we recommend that you refer to the last chapter of our original instruction manual.