Installing and running GEM
Version 5.1.a6

Environment and Climate Change Canada

March 4, 2019

Contents

Chapter 1

Installing GEM

1.1 Required tools and libraries

To compile and run GEM, you will need:

- Fortran and C compilers,
- MPI/OpenMPI library (with development package),
- BLAS library (with development package),
- LAPACK library (with development package),
- basic Unix utilities such as cmake (version 2.8.7 minimum), bash, sed, etc.

If those tools are not available on your computer, you can install them if you have administrative rights. If not, check with your system administrator.

GEM was tested on:

- Fedora Core 29, with gfortran 8.2.1 and cmake 3.12.1
- Fedora Core 29, with ifort 19.0.1 and cmake 3.12.1
- Fedora Core 28, with gfortran 8.2.1 and cmake 3.11.2
- Ubuntu 18.04, with gfortran 7.3.0 and cmake 3.10.2
- Ubuntu 14.04, with gfortran 5.1.0 and cmake 2.8.12.2
- Ubuntu 14.04, with ifort 16.0.1 and cmake 2.8.12.2

For gfortran/gcc, version 5 minimum is needed.

1.2 First steps for the installation of GEM

GEM is ready to be compiled with Intel tools (proprietary compilers available through a license) and GNU Fortran and C compilers (gcc and gfortran: license under the GNU General Public License).

- clone or download the git tar file of GEM at GitLab: http://gitlab.com/gem-ec/goas,
- a directory named *goas*, is created,
- `cd goas`,
- please note in that directory a file named *README*, giving the same information as below.
- execute the script named *download-dbase.sh* or download directly the datafile at the following address and untar it:
  http://collaboration.cmc.ec.gc.ca/science/outgoing/goas/gem_dbase.tar.gz

Make sure the compilers and libraries are in the appropriate *PATHS*: you may have to initialize PATH and LD_LIBRARY_PATH.

Examples for gcc/gfortran (to be changed according to your setup):
- On Ubuntu:
export PATH=/usr/lib/openmpi/bin:$PATH
export LD_LIBRARY_PATH=/usr/lib/openmpi/lib:$LD_LIBRARY_PATH
- On Fedora:
export PATH=/usr/lib64/openmpi/bin:$PATH
export LD_LIBRARY_PATH=/usr/lib64/openmpi/lib$:LD_LIBRARY_PATH

Examples for PGI compilers (to be changed according to your setup):
export PGI=/local/CUDA/PGI/pgi
export LM_LICENSE_FILE=$PGI/license.dat
export PATH=$PGI/bin:$PGI/mpi/openmpi/bin:$PATH
export LD_LIBRARY_PATH=$PGI/lib:$LD_LIBRARY_PATH

Intel compilers:
Make sure you load *compilervars.sh*.

1.3 Compiling and installing GEM

GEM is configured by default to use gfortran and gcc compilers.

- `cd goas/sources/build`
- `cmake ..`

- if you want to compile with another compiler than gfortran, type:
  `cmake -DCOMPILER=ifort ..` or
  `cmake -DCOMPILER=pgfortran ..` or
  edit the *sources/CMakeLists.txt* file to change the default compiler.
- if you get errors messages (for example, compiler or MPI/OpenMPI not found), make sure these tools are exported in their respective *PATHS* (see above)
- if compiler or compile options and flags are not right:
  - remove the content of the build directory: `rm -rf build/*`
  - make changes to the appropriate cmake file, such as :
    sources/Linux-x86_64-gfortran.cmake
  - `cmake ..` (or cmake -DCOMPILER=ifort .. or cmake -DCOMPILER=pgfortran ..)
- `make -j`
- `make install`
- a directory named *work/work-name_of_the_os-name_of_the_compiler/bin* is created in the *work* directory of GEM on a stick, and the following binaries are installed in it: maingemdm.Abs, maingemgrid.Abs, mainyy2global.Abs, flipit.Abs, r.fstinfo, voir, editfst, fststat, cclargs_lite, pgsm.

Chapter 2

Running GEM

Running the GEM model simply consists of setting a few configuration files then running a few scripts that will eventually take care of the three major components of the model execution:

1. Prep: will prepare input date for the model
2. Model: will run the binary maingemdm (main time loop)
3. Output: will run post-processing of model output

2.1 Running the default configuration of the model

In order to run the default configuration of the model:

- `cd goas/work/work*`
- `./runprep` (will create a directory named *PREP*)
- `./runmod` (will run the model and create a directory named *RUNMOD.dir* with output files)
- or use `./runmod -ptopo 2x2x1` (to use 4 cpus for GEM-LAM, 8 cpus for GEM-Yin-Yang - see below for details)

2.2 Tools to inspect the outputs

- Use *voir* to see what records are produced in the FST output files:

  ```
  ./voir -iment RUNMOD.dir/output/cfg_0000/laststep_0000000024/000-000/dm2009042700-000-000_010
  ```

- Use *fststat* to look at statistical means on the records:

  ```
  ./fststat -fst RUNMOD.dir/output/cfg_0000/laststep_0000000024/000-000/dm2009042700-000-000_010
  ```

2.2.1 Tool to visualise the outputs

You can install SPI, a scientific and meteorological virtual globe offering processing, analysis and visualisation capabilities, with a user interface similar to Google Earth and NASA World Wind, developed by Environment Canada.

SPI can be downloaded at http://eer.cmc.ec.gc.ca/software/SPI.
Follow instructions found in *INSTALL.txt*, in the 7.12.0 directory.

2.3 Configuration Files

The execution of all three components of GEM is highly configurable through the use of three configuration files called:

1. gem_settings.nml: file containing some namelists to configure the model execution
2. outcfg.out: file use to configure the model output
3. configexp.cfg: file use to configure the execution shell environment

Examples of these files can be found in the test cases in the *work/configurations* directory.

A fourth configuration file, named physics_input_table, is used for GEM_cfgs, GY_cfgs and LAM_cfgs test cases.

2.4 Running different configurations

The configuration used by default when running the model is in the directory *work/configurations/GEM_cfgs*, but you will find other configurations in the *work/configurations* directory:

- GY_cfgs: using GEM Yin-Yang
- LAM_cfgs: using GEM LAM
- Bubble_cfgs_bubble: theoretical case
- Schaer_cfgs: Schaer mountain wave
- GEM_theo_cfgs: theoretical test cases

You can use them by giving as an option the directory where the configurations files are situated:

To prepare the input: `./runprep -dircfg cfg_dir`
(for example: `./runprep -dircfg GY_cfgs`)

To run the model: `./runmod -dircfg cfg_dir` (for example: `./runmod -dircfg GY_cfgs`)

To run the model on many cpus `./runmod -dircfg cfg_dir -ptopo 2x2x1` (to use 4 cpus for GEM-LAM, 8 cpus for GEM-Yin-Yang - see below for details)

2.5 Running your own configuration

Put the three configurations files (gem_settings.nml, outcfg.out and configexp.cfg) in a directory structure such as: *exp_dir/cfg_0000* in the *work/configurations* directory.

The master directory name (*exp_dir* in the example above) can be any valid directory name. However, the second directory must have the name cfg_0000.

Then run the two scripts with the following commands:

To prepare the input: `./runprep -dircfg exp_dir`

To run the model: `./runmod -dircfg exp_dir`

To run the model on many cpus `./runmod -dircfg exp_dir -ptopo 2x2x1` (to use 4 cpus for GEM-LAM, 8 cpus for GEM-Yin-Yang - see below for details)

2.6 Modifying the grid and getting meteorological data

You can use the script named *grille.sh*, in the *scripts* directory, to define your own grid and visualise it with SPI (see section 2.2.1 above how to get it).

For this, copy one of the *gem_settings.nml* files located in the different configurations directories, edit it, and then run the command `grille.sh -spi`.

If you want geophysical fields and historical meteorological data for the region you defined in that new grid, contact us.

Chapter 3

Other Utilities

3.1 pgsm

pgsm is a utility designed to perform horizontal interpolations and basic arithmetic operations on RPN standard files.

Input files must be RPN standard files. Output files may be RPN standard files (random or sequential), binary FORTRAN sequential files, or formatted ASCII files.

PGSM can:

- Interpolate data on various geographical projections, compressed or not.
- Interpolate wind components UU and VV with respect to the scale and orientation of the output grid.
- Perform symmetric or antisymmetric extrapolations from an hemispheric grid.
- Compute thicknesses between 2 levels.
- Compute precipitation amounts between 2 forecast hours.
- Extract low, mid and high clouds.
- Perform mathematical operations on one field or more.
- Compute latitudes and longitudes from the X-Y coordinates of a grid.
- Read navigational records of latitudes-longitudes (grid type Y) or grid coordinates (grid type Z) in an input or output file and interpolate values from these coordinates.

Example:

```
./pgsm -iment <input FST> -ozsrt <output FST> -i <pgsm.directives>
```

3.2 editfst

editfst is a utility used for editing and copying records from RPN standard files into a new or an existing RPN standard file. It can do a straight (complete) copy of the input file or it can copy records selectively as indicated from the standard input or from a file of directives named in the -i option.

Example:

```
./editfst -s <input FST> -d <output FST> -i <editfst.directives>
```

More details on these utilities will be available soon.