Installing and running GEM
Version 5.2 - Branch dev-5.2

Environment and Climate Change Canada

August 23, 2021

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),
- Portable Hardware Locality library (hwloc) (with development package),
- BLAS library (with development package),
- LAPACK library (with development package),
- fftw3 library (with development package),
- basic Unix utilities such as cmake (version 3.10 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 34, with Intel 2021.1.3 and cmake 3.20.5 (OpenMPI and Intel MPI),
- Fedora Core 34, with GNU compilers 11.2.1 and cmake 3.20.5,
- Ubuntu 18.04, with Intel 19.0.3.199 and cmake 3.10.2
- Ubuntu 18.04, with gfortran 7.5.0 and cmake 3.10.2

For GNU, 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 the git repository:
  `git clone https://github.com/ECCC-ASTD-MRD/gem.git`
  or download the git tar file of GEM at GitHub: https://github.com/ECCC-ASTD-MRD/gem.git,
- a directory named *gem*, is created,
- please note a file named *README.md*, giving the same information as below.

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

Examples for GNU 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
or
export LD_LIBRARY_PATH=/usr/lib/x86_64-linux-gnu/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

Intel compilers:
You may have to change the *MPI_C_COMPILER* and *MPI_Fortran_COMPILER* variables according to your setup, for example if you use OpenMPI instead of Intel MPI library.

1.3 Compiling and installing GEM

GEM is configured by default to use GNU compilers.

- execute the script named *download-dbase.sh*: `./download-dbase.sh .` or, if it does not work, download directly the datafile at the following address and untar it in the main directory:
  http://collaboration.cmc.ec.gc.ca/science/outgoing/goas/gem_dbase.tar.gz
- `cd gem`
- checkout branch dev-5.2: `git checkout dev-5.2`
- get cmake_rpn submodule: `git submodule update -init -recursive cmake_rpn`

- set the environment variables according to the compiler used:
  `. ./.common_setup gnu` or
  `. ./.common_setup intel`
- `mkdir -p build`
- `cd build`
- `cmake ..` (default compiler is GNU suite)
- or if you want to compile with Intel:
  `cmake -DCOMPILER_SUITE=intel ..`
- 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 in the *cmake_rpn/ec_compiler_presets/default* directory:
    Linux-x86_64/gnu.cmake if you compile with GNU or
    Linux-x86_64/intel.cmake if you compile with Intel
  - `cmake ..` (or `cmake -DCOMPILER_SUITE=intel ..`)
- `make -j`
- `make work`
- a directory named after the compiler you used, and the operating system you compiled on (*work-OS_NAME-COMPILER_NAME*: for example *work-Fedora-33_x86-64-gnu-10.2.1*) is created in the main directory of gem, and binaries are installed in a *bin* directory in it.

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 one of the configurations of the model

Configure the model with one of the configurations and execute the model:

- Move to the working directory created, named after the compiler you used and the operating system you compiled on, for example *work-Fedora-33_x86-64-gnu-10.2.1*:
  `cd gem/work-`*`OS_NAME-COMPILER_NAME`*
- `runprep.sh` script will create a directory named *PREP*,
- `runmod.sh` script will run the model and create a directory named *RUNMOD* with output files.
- for example:
  `runprep.sh -dircfg configurations/GEM_cfgs_GY_FISL_P`
  `runmod.sh -dircfg configurations/GEM_cfgs_GY_FISL_P`
- To run the model on several cpus:
  `runmod.sh -dircfg configurations/[cfg_dir] -ptopo 2x2x1`
  (to use 4 cpus for GEM-LAM, 8 cpus for GEM-Yin-Yang)
- If you get an error message saying `runprep.sh` or `gem_dbase` is not found, make sure to set the environment variables using the setup file situated in the main directory:
  `./.common_setup gnu` or `./.common_setup intel`

- default setup uses an in-house script (`r.run_in_parallel`) to run the model. If you want to use another command, or if it doesn't work in your environment, edit the file `scripts/gem_mpirun.sh` to change the script, or move the file `scripts/r.run_in_parallel` so that the model is run with mpirun directly.

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 Tools 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.

You can also install xrec, a visualisation program used to display 2D meteorological fields stored in the RPN standard file format, developed by Research Informatics Services, Meteorological Research Division, Environment and Climate Change Canada. xrec on a stick can be cloned or downloaded at: https://gitlab.com/gem-ec/xoas.

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 *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 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 *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.sh -dircfg configurations/exp_dir`

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

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

2.5 Modifying the grid and getting meteorological data

You can use the script named *grille*, 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 -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>
```