**Project: Michigan Space Weather Modeling Framework**

**Solicitation RA-133W-14-RP-0132**

Milestone 4.1.1

Deadline: January 30, 2015

**Documentation of Algorithms**

1. **Installing and Using the Geospace Model**

**1.1 Code version**

The Geospace model is part of the Space Weather Modeling Framework (SWMF). The SWMF repository is stored on the herot.engin.umich.edu machine. We created a user account for George Millward on herot. To access the CVS repository, the CVSROOT environment variable has to be set to the value

gmillward@herot.engin.umich.edu:/CVS/FRAMEWORK

Once the CVSROOT is set (and of course cvs has to be installed) the HEAD (i.e. current) version of the SWMF can be checked out with

cvs checkout SWMF

The initial version of the Geospace model was transferred to SWPC on January 23, 2014. For sake of stability, initially we are using this version with small modifications instead of the HEAD version. This branch of the SWMF can be checked out with

cvs checkout –r SWPC\_2014\_01\_23 SWMF

After CVS checkout the code will be in the SWMF directory. This directory can be moved and/or renamed before installation. Once the code is installed, the directory should not be moved or renamed (unless it is uninstalled). In the rest of this document, unless otherwise stated, it is assumed that the current directory is the main SWMF directory.

We have also provided access to the Linux machine roprop.engin.umich.edu where the various versions of the code can be found in the

/data2/SWPC/SWMF\_HEAD/

/data2/SWPC/SWMF\_01\_23\_2014/

directories.

Note that up to now we have implemented changes and new features both in the SWPC\_2014\_01\_23 branch and the HEAD version of the SWMF, so that future transitions to newer versions of the SWMF will be smooth. Since many further changes are planned, we decided to switch to the HEAD version so the new features are only implemented once. Our intent on the long run is to maintain at most two versions of the SWMF in the future: the one running at SWPC and the HEAD version.

**1.2 Installation, Configuration, Compilation**

The SWMF needs to be installed first with the appropriate Fortran compiler selected. Since currently SWPC uses the Intel Fortran compiler the installation can be done with

./Config.pl –install -compiler=mpiifort,icc

If a new version of the SWMF and/or compiler is used, it is best to do a short test. Assuming that the code can run with the “mpirun” command on the machine,

make –j test3 MPIRUN=’mpirun –np 4’

will execute a short test on 4 cores and compilation will be done in parallel due to the –j flag. The expected output is two empty files for GM and IE and small errors for IM:

-rw-r--r-- 1 gtoth hpcc 0 Jan 16 10:18 test3\_gm.diff

-rw-r--r-- 1 gtoth hpcc 0 Jan 16 10:18 test3\_ie.diff

-rw-r--r-- 1 gtoth hpcc 16578 Jan 16 10:18 test3\_im\_rcm.diff

Running test3 takes care of configuring the SWMF the way it is used for the Geospace model. In fact

make test3\_compile

can be used to configure and compile the SWMF for the Geospace model. But it is probably better to describe what is happening under the hoods, so that the configuration of the SWMF for the Geospace model is clear.

The SWMF has to be configured with three components for the Geospace Model: the Global Magnetosphere model (BATS-R-US) in the GM/BATSRUS directory, the Inner Magnetosphere model (RCM) in the IM/RCM2 directory and the Inner Electrodynamics model (RIM) in the src/Ridley\_serial directory:

./Config.pl –v=Empty,GM/BATSRUS,IM/RCM2,IE/Ridley\_serial

The first “Empty” value means that all other models are switched off first (in case the SWMF was configured already) and then the three specified components are switched on. BATS-R-US has many options that can be configured. The most important are the grid and the user and equation modules. These can be configured as

./Config.pl -g=GM:8,8,8,400,1 -o=GM:u=Default,e=Mhd

The –g flag sets the size of the grid blocks to 8x8x8 cells, the maximum number of grid blocks to 400, and 1 of these can be advanced with the part implicit time stepping scheme (setting the last value to 1 means that the part implicit time stepping will not be used). The u= flag sets the user module to Default (nothing special) and the e= flag sets the equation module to Mhd for BATSRUS. There are many other options that can be configured for each model (as can be seen by typing e.g.

GM/BATSRUS/Config.pl –h

but the default settings are appropriate for the Geospace model. The current configuration is shown if Config.pl is run without arguments.

The optimization level of the compiler affects execution speed, but also accuracy and reliability. We have been using the –O2 optimization level for many years with earlier versions of the ifort compiler, because the –O3 did not work properly. More recently, we found that the –O3 optimization level works well with the newer versions of the ifort compiler. We have modified the code to allow –O3 optimization, so the default optimization level is now –O3. The optimization level can be set back to level 2 with

./Config.pl –O2

We recommend performing comprehensive testing whenever the compiler version and/or the compilation level are changed. As a minimum, test3 should be rerun.

After the configuration is complete, the SWMF can be compiled with

make –j SWMF

where the optional –j speeds up the compilation by using multiple cores in parallel. Some of the output files may require post-processing, so it is best to compile the appropriate code with

make PIDL

After successful compilation the executables will be in the bin/ directory

bin/SWMF.exe

bin/PostIDL.exe

If some of the source files are modified (e.g. a new feature is installed) the code can be updated with

cvs update

or specific file or directory can be updated, for example

cvs update GM/BATSRUS/src

After an update, the code can be recompiled without cleaning the existing object files.

If necessary (e.g. to change the optimization level or to test a different compiler version), the compiled executables and object files can be removed with

make clean

prior to a recompilation. The code can be uninstalled (e.g. to move it to a different directory) with

./Config.pl -uninstall

**1.3 Setting up the run directory**

The SWMF in general (and the Geospace model in particular) runs in a directory that is dedicated for the run and separate from the CVS distribution. This directory contains several files and subdirectories, so it has to be created with the SWMF (after the code is configured) with

make rundir

The resulting run/ directory can now be renamed or moved into a different location if desired. Note, however, that the directory contains symbolic links to the executables. One can replace the links with copies of the executables if desired, e.g.

rm –f run/SWMF.exe; cp bin/SWMF.exe run/

rm –f run/GM/PostIDL.exe; cp bin/PostIDL.exe run/GM/

If the run directory is moved and the links are removed, it may be a good idea to add some file into it that indicates where the corresponding source code can be found. For sake of simplicity we keep using “run/” as the name of the run directory.

There are a number of input files that need to be put into the run directory. Two of these have fixed names and are required for any run with the SWMF:

run/PARAM.in

run/LAYOUT.in

The PARAM.in file is different for initial startup and continuation of the run using restart files. The initial startup should be done only once and the recommended file is in

Param/SWPC/PARAM.in\_SWPC\_init

that should be copied into run/PARAM.in for initial startup. For restarted runs, which will be all the runs except for the very first one, the

Param/SWPC/PARAM.in\_SWPC\_restart

file should be used.

The LAYOUT.in file specifies which component runs on which processor. For the 64 core runs the

Param/SWPC/LAYOUT.in\_SWPC

file can be used as-is (IM runs on processor 0, IE runs on processors 0 and 1, and GM runs on processors 2 to 63 or the maximum processor index in general).

There are additional files that are needed for running the Geospace model in operational mode. The names of these files are defined in the PARAM.in file, so their names can be modified. For sake of simplicity here we use the customary names to refer to the various files. The

IMF.dat

file contains the solar wind and interplanetary magnetic field data propagated to the inflow boundary of the global magnetosphere domain (typically at x=32Re in the GSM coordinate system). The IMF file is a simple ASCII file that should have the following format:

#START

2006 12 14 7 0 0 0 0.00 -3.01 -0.56 -585.89 -11.35 1.73 1.39 171087.25

2006 12 14 7 1 0 0 0.00 -2.82 -0.82 -583.52 -12.21 1.25 1.43 172988.88

2006 12 14 7 2 0 0 0.00 -2.77 -0.77 -580.37 -13.88 -1.11 1.56 172606.37

…

where the “#START” indicates the start of the solar wind data containing 15 space separated columns per line. The first 7 columns are integers containing the date and time from year down to milliseconds in coordinated universal time, UTC. The next three columns contain the 3 components of the interplanetary magnetic field BSW in nT, the next 3 columns are the 3 components of the velocity vSW in km/s, the next column is the number density nSW in cm-3, and the last column is the ion temperature TSW in K. The vector quantities are in the GSM coordinate system by default (this can be modified as described in the GM/BATSRUS/PARAM.XML file for the #SOLARWINDFILE command). Since BATS-R-US runs in single fluid MHD mode in the Geospace model using protons and negligible electron temperature, the mass density will be the number density times the proton mass, and the thermal pressure will be pSW = nSWkTSW. **Important notes:**

1. The X component of the magnetic field should not be set to zero (unlike the above example).
2. For proper linear interpolation in time the file should contain at least two data rows.
3. The time series should be monotonically increasing, but it does not have to be uniformly spaced.

The IMF file can be continuously updated with new data lines if the PARAM.in file contains the newly added command

#REFRESHSOLARWINDFILE

T DoReadAgain

The effect of this command is that the code stops running when the simulation reaches the time of the final data point and waits until the solar wind file gets updated. The update can be a simple append to the end, or it can be a complete replacement of the file too. Currently there is no locking mechanism to make sure that the code does not try to read a partially written solar wind file. Although this has not caused any issues so far, we may have to do that to improve the robustness of the code.

Another input file contains the coordinates of the magnetometer stations, an example can be found in

Param/SWPC/magin\_GEM.dat

There can be several input files containing satellite trajectories so that synthetic in-situ data can be extracted from the simulation. The satellite trajectory files should be produced in advance (unlike the solar wind file) and read at the beginning of the simulation. An example for defining several trajectory files is provided in

Param/SWPC/SATELLITES.in

that can be copied into the run directory, modified as needed, and included into the PARAM.in file with the

#INCLUDE

SATELLITES.in

command. See the description of the #SATELLITEFILE command in GM/BATSRUS/PARAM.XML for more detail.

**1.4 Running the Geospace model**

There are some important differences between running the Geospace model in *research mode* (which includes testing and validation studies as well) and in *real-time mode*. In research mode the input is prepared, the code is typically run only once and the output is post-processed and analyzed after the run finished. In real-time (operations) mode the inputs are updated while the code is running continuously and the output is also post-processed and used at the same time. We have already developed most of the tools needed for real-time mode, but some of these can and will be further optimized and customized as part of the collaboration between SWPC and UofM.

In operation mode the first step is to create an initial state. The run/PARAM.in file should be based on the Param/SWPC/PARAM.in\_SWPC\_init file with the following modifications:

* The #STARTTIME should be set to the time (in UTC) when the time dependent simulation will be started.
* The #CPUTIMEMAX command should be removed (or the maximum time should be set to a value that is consistent with the planned periodic daily restarts, in seconds).
* The F10.7 flux should be set to the current (or very recent) value in the #IONOSPHERE command.

In addition the solar wind data file (IMF.dat) should contain at least 2 data lines with the current (or very recent) solar wind conditions. Once the input is all set the run can be started as

mpirun –np 64 SWMF.exe > runlog \_`date +%y%m%d%H%M`

or similar command in the run directory. This may be done from a job script.

In addition to running the SWMF.exe, there are a few scripts that need to run. The output generated for the 2D cut files (in run/GM/IO2/) should be continuously post-processed with

PostProc.pl –r=10 >& PostProc.log &

The –r=10 flag instructs the script to check for new output every 10 seconds. The PostProc.log file contains information about the host machine that the process is running on, the start time and the expected end time. This script runs the PostIDL.exe code parallel on 4 cores by default. The number of cores can be changed (see PostProc.pl –h for various options). Note that the script stops after 2 days if not killed sooner.

In operation mode the Geospace model will be restarted at least once a day (regular restart) and whenever the solar wind speed increases suddenly. This means that the SWMF has to save restart files quite frequently (we plan to do that once a minute of simulation time) and the restart files cannot be overwritten (which is the default behavior). We added a new feature controlled by the #RESTARTOUTDIR command that creates a new restart directory every time with a name containing the date and time in the YYYYMMDD\_HHMMSS format.

**1.4 Regular Restart of the Geospace model**

For the daily restart the code can stop simply because it reaches the final simulation time that is set in the #ENDTIME command. Alternatively a clean stop can be achieved by creating the SWMF.STOP file by

touch SWMF.STOP

in the run directory. The existence of this file is checked periodically if the #CHECKSTOP command is present in the PARAM.in file. Checking the file very frequently can slow down the progress of the code. Once the file is noticed, the code will generate a final restart file and stop.

For restart, the PARAM.in file should be replaced with the version that can do a restart and is based on the Param/SWPC/PARAM.in\_SWPC\_restart file. It should contain the most up-to-date F10.7 flux. The solar wind data file (IMF.dat) should be updated too.

If the Restart.pl script is running continuously in the background with the –r flag, the SWMF will link to the last restart directory. If the restart trees are created by the SWMF.exe, it will be necessary to select the last restart directory with

Restart.pl –i SWMF\_RESTART.YYYYMMDD\_HHMMSS

If the Geospace model is run with repeated restarts indefinitely, the number of time steps and the simulation time could accumulate to values that would cause issues. To avoid this problem, the PARAM.in file uses the #ENDTIME command instead of the usual #STOP command in the last session. If the code stops because it reached the time specified with the #ENDTIME command, the SWMF restart file (SWMF\_RESTART.YYYYMMDD\_HHMMSS/RESTART.out) will be written with 0.0 simulation time in the #TIMESIMULATION command and 0 time step in the #NSTEP command, but the #STARTTIME command will have the date and time specified in the #ENDTIME in PARAM.in. The restart file of BATSRUS will still accumulate the time steps, so this issue needs to be fixed.

Now the Geospace model can be restarted with the usual command:

mpirun –np 64 SWMF.exe >& runlog\_`date +%y%m%d%H%M`

**1.4 Restarting the Geospace Model due to Sudden Increase of the Solar Wind Velocity**

Most restarts will be needed, because the solar wind speed increases, so that the propagation of the latest solar wind data from the L1 point to the upstream boundary will take less time than assumed previously. This means that the Geospace model has to go back to an earlier time and use the updated solar wind data, i.e. it has to restart from an earlier restart file.

First of all the running code should be stopped. This may be easiest to do with some Unix command (kill) or from the queue system software (e.g. qdel).

For restart, the PARAM.in file should be replaced with the version that can do a restart and is based on the Param/SWPC/PARAM.in\_SWPC\_restart file. It should contain the most up-to-date F10.7 flux. The solar wind data file (IMF.dat) should be updated too. The restart directory with the new propagated date-time value should be selected:

Restart.pl –i SWMF\_RESTART.YYYYMMDD\_HHMMSS

All the newer restart directories should be removed. The post processing script PostProc.pl should be stopped (killed) to avoid processing files that are being removed. Then the output that is newer than the new date-time should be removed (or archived into a different directory, if desired). This cleanup operation will require a new script or a new option for the Restart.pl script.

Now the Geospace model and the post processing code can be restarted with the usual commands:

mpirun –np 64 SWMF.exe >& runlog\_`date +%y%m%d%H%M`

PostProc.pl –r=10 >& PostProc.log &

**2. Solar Wind Propagation**

Solar wind data arrives near real time to SWPC. This information is measured at the L1 point that is approximately 0.01 AU from the Earth towards the Sun. Let us denote the distance between the satellite and the upstream boundary of the Geospace model with

D = xSAT – xBOUNDARY

where x is the GSM coordinate parallel with the Sun-Earth line. In the current model setup xBOUNDARY = 32 RE = 204,096 km, while xSAT should be obtained from the satellite trajectory. At any given time one can assume that the solar wind propagates with the measured velocity VX(t) so it will arrive to the boundary of the Geospace model domain at

Clearly t’ will be in the future which provides the predictive capability for the Geospace model. In practice the solar wind data is taken as a discrete time series with some uniform frequency Δt, so tn = n Δt and

The solar wind data file will contain the date and time corresponding to together with the measured solar wind parameters. This will work as long as is a monotonically increasing series, i.e.

.

As long as does not change rapidly, this will be true. However, if

then . Note that is the ratio of the measurement frequency and the time it takes for the solar wind to propagate from L1 to the boundary, which is much less than 1 if the measurements are taken frequently enough.

Whenever the above condition is met, the Geospace model will have to be restarted from a time that is less than and all simulation results (and predictions) beyond this time should be discarded.

**3. Explicit versus Implicit Time Stepping Schemes**

The following discussion is about the time stepping schemes used in BATSRUS. Since BATSRUS dominates the execution time for the Geospace application, the choice of the time integration scheme in BATSRUS has a major impact on the overall performance of the SWMF.

Explicit time integration uses the current state to calculate fluxes and source terms, and then uses these to advance the solution to the next time step. This method is fast but the time step is limited by numerical stability conditions, which make the time step small (about 0.1s or less for the magnetospheric runs using the typical grid resolution).

Implicit time integration uses fluxes and sources based on the already advanced state. This requires a solution of a large linear system, which is expensive. On the other hand the implicit time integration is stable for large time steps. In the SWPC application we typically try to use 5 second time steps.

The part implicit scheme uses the explicit time stepping in the grid blocks where the time step is stable, and the implicit time stepping where the explicit method would be unstable. The stability condition is evaluated every time step.

The time step may be reduced if the density or pressure changes too much at any location in the grid. This can happen both for the explicit and implicit time stepping, but it is more likely to happen for the implicit scheme, because the larger time steps result in larger changes in general.

When the time step is reduced, the part implicit scheme switches more-and-more blocks from implicit to explicit. If the time step gets very small, all blocks may become explicit. If the changes in density and pressure become smaller, the code gradually tries to recover the original time step.

We have 1-stage and 2-stage explicit time stepping schemes. In the explicit 1-stage scheme the fluxes and source terms are calculated from the current state and then multiplied with the time step Dt and added to the current state to get the new state. In the 2-stage scheme, on the other hand, we first advance the solution by a half time step Dt/2 and then use this intermediate state to recalculate the fluxes and sources, and then use these to update the solution with a full time step Dt.

The 1-stage scheme has a temporal discretization error proportional to Dt, while the two stage scheme makes the error term proportional to Dt2. The 2-stage scheme is about twice as expensive as the 1-stage scheme. For the explicit time stepping scheme the time step is small, so the temporal error term is small compared to the spatial discretization error that is typically proportional to Dx2 where Dx is the cell size.

Based on several experiments, we now recommend using the 1-stage explicit time stepping scheme for operational use, as it can continuously maintain faster than real time performance on 64 processor cores while the results remain essentially the same as with the original implicit time stepping scheme. The PARAM.in files in the Param/SWPC directory were modified accordingly.