Global Modeling Initiative
Users' Guide I

Jules Kouatchou, Bigyani Das and Jae Kim
NASA/GSFC Code 931
Greenbelt, MD 20771
{*kouatch, bdas, jkim*}*@calvin.gsfc.nasa.gov*

April 22, 2004

Abstract

This tutorial is written to help new users to install and test the Global Modeling Initiative (GMI) code. We provide specific instructions on how to obtain the code, to properly set environment variables, to select the model configuration, to choose a particular platform, to compile the code and to perform basic test runs. The focus of the document is for the installation and execution of the GMI code on `halem`, `daley` and `thunderhead`.

Contents

1 The Global Modeling Initiative **2**
1.1 Overview of the Model 2
1.2 Components of GMI 2

2 Installation and Testing **3**
2.1 Getting the Code 3
2.2 Model Files and Directory Structure 3
2.3 Setting Environment Variables 4
2.4 Code Installation and Basic Test Run 7

3 Input Files, Met Fields Data Files and Output Files **10**
3.1 Input Files or Namelist Files 10
3.2 Met Fields 11
3.3 Output Files 11

4 Specific Runs **12**

5 Summary of the Necessary Steps **14**

1 The Global Modeling Initiative

1.1 Overview of the Model

The Global Modeling Initiative (GMI) [1] was initiated under the auspices of the Atmospheric Effects of Aircraft Program (AEAP) in 1995. The goal of GMI is to develop and maintain a state-of-the-art modular 3-D chemistry and transport model (CTM) that can be used for assessment of the impact of various natural and anthropogenic perturbations on atmospheric composition and chemistry, including, but not exclusively, the effect of aircraft.

More recently, ACMAP has selected the approach of GMI to serve as both and assessment facility and a testbed for model improvements for future assessments in all areas of atmospheric chemistry.

The GMI model is a modular chemical-transport model (CTM) with the ability to carry out multi-year assessment simulations as well as incorporate different modules, such as meteorological fields, chemical mechanisms, numerical methods, and other modules representing the different approaches of current models. This capability facilitates the understanding of the differences and uncertainties of model results.

The testing of GMI results against observations is a high priority of GMI activities. Science Team members contribute by either supplying a particular module and/or contributing to the analysis of the results and comparison with atmospheric observations [2, 4].

Besides acting as a testbed for different modules, GMI will also act as a 3-D assessments assessment facility. The GMI modular code is currently implemented at NASA/Goddard Space Flight Center (the core institution). The core institution is responsible for: integrating and testing components of the GMI model, maintaining coding standards which will make the model transportable to different platforms, carrying out assessment calculations, and providing first-order results and diagnostics for analysis by team members. The current code has been developed to be run on a variety of computing platforms, both with single and multiple processors. This report provides instructions on how to install and execute the GMI code on computing platforms available at NASA/Goddard Space Flight Center or in particular at the NASA Center for Computational Science (NCCS) [2].

More information on GMI can be obtained in [1, 2, 3, 4].

1.2 Components of GMI

The modules that make up the GMI assessment model are [4]:

1. Input meteorological data coming from three major Global Circulation Models (see Section 3.2 for details)

2. Advection algorithm

3. Mass tendencies

4. Numerical schemes for chemistry solutions

5. Chemistry mechanism

[1] http://gmi.gsfc.nasa.gov/gmi.html
[2] http://nccs.nasa.gov

6. Heterogeneous processes
7. Photolysis
8. Diagnostics
9. Tropospheric treatment
10. Initial conditions
11. Boundary conditions

All the above modules have multiple options.

2 Installation and Testing

During this process of installing and testing the code, it is assumed that *Cshell* is the default shell employed by the user. In fact, the GMI environment variables required for these procedures are set up using *Cshell*.

2.1 Getting the Code

To obtain the GMI code,

- Select the directory where you want to install the GMI model, say *MYGMI/*
- Get the model from *sourcemotel* by typing the command lines:

```
%setenv CVS_RSH ssh
%cvs -d usrid@sourcemotel:/cvsroot/gmi co gmi_gsfc
```

Here *usrid* is your login name on `sourcemotel`. You will be asked to provide your password on `sourcemotel`. The directory *gmi_gsfc/*, which is the main GMI directory, will then appear.

2.2 Model Files and Directory Structure

Move into *gmi_gsfc/*

```
%cd gmi_gsfc
%ls
```

You will find (in *gmi_gsfc/*) the files and directories:

```
CVS/          README.install    gem/         gmi_install
README.first  cshrc.ggmi        gmi_data/    login.ggmi
```

The top directory of the GMI code is *gem/* which contains sub-directories presented in Tables 1, 2 and 3.

Subdirectory	Description
actm/	Atmospheric Chemistry Transport Model
actm/gmimod/	
actm/gmimod/Other/	
actm/gmimod/Other/doc/	README files
actm/gmimod/Other/misc/	
actm/gmimod/Other/scripts/	Script files performing various functions.
actm/gmimod/Other/test/	Basic test cases to run the model.
actm/gmimod/Other/test/nopar/	
actm/gmimod/Other/test/nopar/Old/	
actm/gmimod/Other/test/nopar/infiles/	Input namelist files
actm/gmimod/Other/test/nopar/outfiles/	ASCII ouput files for test cases in *infiles/*
actm/gmimod/Other/test/par/	
actm/gmimod/Other/test/par/infiles/	Input namelist files
actm/gmimod/Other/test/par/outfiles/	ASCII ouput files for test cases in *infiles/*
actm/gmimod/advec/	
actm/gmimod/advec/dao2advec/	
actm/gmimod/advec/dao2utils/	
actm/gmimod/advec/include/	
actm/gmimod/chem/	Subdirectories for chemestry calculations.
actm/gmimod/chem/aerosol/	
actm/gmimod/chem/aerosol/include_setkin/	
actm/gmimod/chem/aerosol/setkin/	
actm/gmimod/chem/include/	
actm/gmimod/chem/sad/	
actm/gmimod/chem/smv2chem/	
actm/gmimod/chem/stratosphere/	
actm/gmimod/chem/stratosphere/include_setkin/	
actm/gmimod/chem/stratosphere/setkin/	
actm/gmimod/chem/sulfur/	
actm/gmimod/chem/troposphere/	
actm/gmimod/chem/troposphere/include_setkin/	
actm/gmimod/chem/troposphere/setkin/	

Table 1: GMI directory structure I.

2.3 Setting Environment Variables

From the directory *gmi_gsfc*, read all the README files by starting with *README.first* file that guides a new user to take the required steps for installing and running the GMI code. The top portions of the files *cshrc.ggmi* and *login.ggmi* include instructions for setting up the environment variables.

Edit the file *cshrc.ggmi* (for the architecture you want the GMI model to run on) and update

Subdirectory	Description
actm/gmimod/comm/	MPI communication subroutines.
actm/gmimod/control/	
actm/gmimod/convec/	
actm/gmimod/depos/	
actm/gmimod/depos/include/	
actm/gmimod/diffu/	
actm/gmimod/emiss/	
actm/gmimod/emiss/harvard/	
actm/gmimod/emiss/include/	
actm/gmimod/emiss/llnl/	
actm/gmimod/in_out/	Subroutines for input/output.
actm/gmimod/include/	Include files.
actm/gmimod/include_data/	
actm/gmimod/mem_manage/	Routines for dynamic memory allocation
actm/gmimod/phot/	
actm/gmimod/phot/fastj/	
actm/gmimod/phot/include/	
actm/gmimod/phot/lookup/	
actm/gmimod/phot/utils/	
actm/gmimod/step/	The time stepping routine *gmi_step.F* and other important routin
actm/gmimod/trans/	
actm/gmimod/trans/include/	
actm/gmimod/trans/ucitrans/	
actm/gmimod/trans/ucitrans/Uci_Data/	
actm/lib/	

Table 2: GMI directory structure II.

the variables *GEMHOME* (location of the main model directory) and *GMI_DATA* (directory where the input data are located):

```
setenv GEMHOME   ~/MYGMI/gmi_gsfc/gem
setenv GMI_DATA  ~/MYGMI/gmi_gsfc/gmi_data
```

In addition, include the variable *CHEMCASE* used to select the chemistry routine for the assigned GMI model, currently *troposphere*, *aerosol*, and *stratosphere*. Write the following line to choose for instance *stratosphere*:

```
setenv CHEMCASE stratosphere
```

Copy the files *cshrc.ggmi* and *login.ggmi* to your home directory

```
%cp cshrc.ggmi ~/.cshrc.ggmi
%cp login.ggmi ~/.login.ggmi
```

Subdirectory	Description
bin/	Location where the executable is placed
doc/	General information about *gem/* and how it is to be used.
esm/	All of the Earth System Model -level source code and include files. Routines in this directory control and coordinate the functioning of all of the other packages.
esm/comm/	MPI communication routines
esm/control/	Routines controling esm functions
esm/in_out/	
esm/include/	
esm/lib/	
esm/main/	Main program *esm_main.F*
esm/mem_manage/	
esm/utils/	
esm_tools/	All of the esm_tools source code and include files
esm_tools/baseline/	
esm_tools/fi/	
esm_tools/include/	
esm_tools/lib	
include/	General purpose header files that are used throughout the code. The files contain compilation, platform selection options, etc.

Table 3: GMI directory structure III.

Go to the directory *gem/include/* and edit the file *gem_sys_options.h*. Modify the line

```
#define ARCH_OPTION  ARCH_XXXX
```

to select the architecture you want to run the code on. *XXXX* is *COMPAQ* for `halem`, *SGI_ORIG* for `daley`, and *INTEL* for `thunderhead`.

You may also choose to edit the file *gem_options.h* to select debugging, optimization, or profiling options. If necessary, provide the paths to MPI and netCDF include files and libraries in the file *gem_config.h*. Some compilation options may have to be changed in the same file. Here are some necessary changes in *gem_config.h*:

```
For ARCH_SGI_ORIG
-----------------
     INCLUDES_NETCDF = -I/usr/local/include/netcdf_64
     CC          = cc
     FC          = f90

For ARCH_INTEL
--------------
     INCLUDES_NETCDF = -I/usr/local/netcdf/include
     NETCDFLibs      = -L/usr/local/netcdf/lib -lnetcdf
```

```
FC          = mpif90
FFLAGS      = -c -extend_source -cm -w95 -WB  \
              $(DEBUG_FLAG) $(OPTIMIZE_FLAG) $(PROFILE_FLAG)
```

Go to your home directory and edit the file *.cshrc*

```
%cd ~/
%vi .cshrc
```

Include the lines

```
setenv CVS_RSH ssh
setenv ARCHITECTURE  ARCH_XXXX
if (-e ~/.cshrc.ggmi) then
   source ~/.cshrc.ggmi
endif
```

Update the changes made in the *.cshrc* file

```
%source .cshrc
```

You can also edit the *.login* file and add the lines

```
if (-e ~/.login.ggmi) then
    source ~/.login.ggmi
endif
```

Update the changes by typing

```
%source .login
```

2.4 Code Installation and Basic Test Run

Go back to the working directory *MYGMI/gmi_gsfc/gem/*:

```
%cd gem
```

Compiling the model

To compile the code on `thunderhead` you need to select the proper MPI environment variables. To view what is available, type

```
%mpi_env -p
```

Since we want to run with *intel*, type

```
%mpi_env -c intel
```

to change the MPI environment to *intel*.

To compile the code on all the platforms, use the commands:

```
%mkmf
%make
```

A binary file named *gem* will be created in *$GEMHOME/bin/*.

Remark 1 *You may face problems compiling the code on* `daley` *and* `thunderhead`*. The first one is due to the fact that the f90 compiler on both systems complains that lines in some Fortran files use more than 72 columns. This is strange because the compilations options are properly set.*

The second problem is due to the presence of MPI-2 calls in the code. The linking fails because of those calls. The calls occur in the files esm/comm/msg3d.F, esm/control/esm_generate.F, *and* esm/control/esm_terminate.F. *You need to introduce* #ifdef *statements to remove the MPI-2 (when do_mpi2_border = 1) calls at preprocessing. In addition, you need to edit the file* esm/comm/Makefile.cpp *to prevent the file* mpi2_bord_comm.F *(driver for MPI-2) from being compiled.*

Testing the executable

To test the executable, we will use a sample namelist file coming with the code. For each platform, we provide job script files (named *gmitest.job*) to test the executable. On `halem` and `daley` you need to have your sponsor code account (use the command `getsponsor` to obtain it).

It is assumed that the user wants to test the model from the directory */scratch/usrid* on `daley`, */scr/usrid* on `halem` and */mnt/pvfs1/pvfs1/usrid* on `thunderhead`[3].

```
gmitest.job on daley
      #!/bin/csh -fx
      #PBS -N gmi_COrun
      #PBS -l ncpus=16
      #PBS -l walltime=00:05:00
      ##PBS -l mem=2gb
      #PBS -A a930b
      #PBS -S /bin/csh
      #PBS -V
      #PBS -e error.file
      #PBS -o output.file
      #
      cd /scratch/usrid
      cp $GEMHOME/actm/gmimod/Other/test/par/infiles/dao2rn_dao46_06.in .
      ln -s $GMI_DATA gmi_data
      #
      mpirun -np 16 $GEMHOME/bin/gem -d dao2rn_dao46_06.in

gmitest.job on halem
      !/bin/csh
      #BSUB -P a930b
      #BSUB -J gmitest
      #BSUB -n 16
```

[3]It is important to note that the model will only run on `thunderhead` from *pvfs*. In addition, the MPI commands work only with bash shell. Please refer to http://newton.gsfc.nasa.gov/thunderhead/ for details.

```
#BSUB -W 00:20
#BSUB -q general_lng
#BSUB -o gmiout.%J
#BSUB -e gmierr.%J
#
cd /scr/usrid
cp $GEMHOME/actm/gmimod/Other/test/par/infiles/dao2rn_dao46_06.in .
ln -s $GMI_DATA gmi_data
prun -n 16 $GEMHOME/bin/gem -d dao2rn_dao46_06.in
```

Remark 2 *Replace a930b in the above script files with your sponsor code account. Here* usrid *is the user's login name.*

```
gmitest.job on thunderhead
      ### Number of nodes, processes and minutes requested
      Nno=8
      Npr=16
      Nmi=60
      ltmbegin -n $Nno -m $Nmi
      #
      WORK_DIR=/home/usrid/MYGMI/gmi_gsfc/gem
      RUN_DIR=/mnt/pvfs1/pvfs1/usrid
      cd $RUN_DIR
      cp $WORK_DIR/bin/gem /home/usrid/rhome/MYGMI/gmi_gsfc/gem/bin
      cp $WORK_DIR/actm/gmimod/Other/test/par/infiles/dao2rn_dao46_06.in .
      ### Submit the job
      time ltmpi -n $Nno -t $Npr=16 $WORK_DIR/bin/gem -d dao2rn_dao46_06.in
      ### End the session
      ltmend
```

To submit the job script, do the following

```
On daley
    %qsub gmitest.job
On halem
    %bsub < gmitest.job
On thunderhead
    %mkdir /home/usrid/rhome/MYGMI /home/usrid/rhome/MYGMI/gmi_gsfc
    %mkdir /home/usrid/rhome/MYGMI/gmi_gsfc/gem
    %mkdir /home/usrid/rhome/MYGMI/gmi_gsfc/gem/bin
    %bash
    %ltmsuper
    %./gmitest.job
```

Remark 3 *Though the* `thunderhead` *job script can only be submitted from the PVFS directory, the executable should not reside there. It is suggested to create the same directory structure (location of the executable) on the development node and the execute nodes. This explains the presence of the the multiple* `mkdir` *commands.*

When the job is complete, compare the new output result with the previous result:

```
diff dao2rn_dao46_06.asc $GEMHOME/actm/gmimod/Other/test/par/outfiles/dao2rn_dao46_06.asc
```

The matching results ensure the installation and compilation of GMI model is complete.

Remark 4 *Due to the changes made within the code, it may turn out that the verification process fails at the end of the run. That does not mean the installation of the code was unsuccessful. A failure may be due to the fact that the latest version of the code contains several updates (as a result of the bugs that were found). The comparison is done between the output from the new code and the output of the original one (containing bugs).*

o far we have described the process of installation of the GMI code. Now we want to present how to run special test cases. Before that, we introduce in the next section the input/output files needed/expected for a given run.

3 Input Files, Met Fields Data Files and Output Files

3.1 Input Files or Namelist Files

In the *gem/actm/gmimod/Other/test/par/infiles/* directory, sample namelist input files for different types of runs are available. They are (check the *README.namelist* file in the directory *gem/actm/gmimod/Other/doc/* for more information):

- *dao2be_koch_ncar52_10.in*: the input file to make runs for beryllium. This uses Koch table for Be-7 and Be-10.
- *dao2be_nagai_ncar52_10.in*: used to make runs for Be-7 and Be-10 using Nagai table.
- *dao2co2_dao29_10.in*: for CO runs
- *dao2coldiag_ncar52_10.in*: for Beryllium runs
- *dao2isop_dao46_02.in*: used to make runs for Isoprene, Acetone, Propene and NO
- *dao2n2o_dao29_10.in*: for N2O runs
- *dao2n2o_dao44_10.in*: for N2O runs
- *dao2no_dao46_02.in*: for NO runs
- *dao2rn_dao46_06.in*: used to make runs for radionuclides Pb and Rn
- *dao2sf6_dao29_10.in*: for sulfur runs
- *gmit4_giss23_06.in*: for full chemistry runs
- *pfixllnl_dao29_04.in*: used to make runs with LLNL pressure fixer algorithm
- *pfixuci_dao29_04.in*: used to make runs with GMI pressure fixer algorithm (Prather's algorithm)

To run similar cases with GISS met fields and CCM3 met fields one can use the appropriate naming convention and then modify the input files to reflect the corresponding cases.

Additional information on namelist variables can be obtained in the *Get_Namelist* routine in: *gem/actm/gimimod/in_out/gmi_namelist.F*.

- For example when someone wants to make a radionuclide run for Pb and Rn, then he/she can copy the file *dao2rn_dao46_06.in* to *gissrn_giss23_06.in* and then change the number of levels from 46 to 23 (`k2_gl = 23`), use GISS `met_infile_names` instead of DAO `met_infile_names`, etc.
- If someone wants to run for a year instead of six hours, he can change `tfinal_days` to 355.
- If someone wants to print out monthly averages, then they have to use `do_mean=T` and `pr_nc_period_days = -1.0d0`.

Note that exact path for `gmi_data_dir` and for other input files such as `met_infile_names`, `fixed_const_infile_name`, etc. are needed.

3.2 Met Fields

Met fields from 3 different global models are used for GMI simulations. They are: DAO, GISS and CCM3. The files (located in the directory *MYGMI/gmi_gsfc/gmi_data/*) that come with the code are

```
(DAO Met Fields)
     DAO_GS_4x5x29_951101.nc
     DAO_GS_4x5x46_971023.nc
(GISS Met Fields)
     GISS_2prime_4x5x23_770701.nc
(CCM3 Met Fields)
     NCAR_CCM2_4x5x44_961115.nc
     NCAR_MATCH_4x5x52_970101.nc
(Miscellanous files)
     GISS_gmit3_sad.nc
     fixed_acetone_gmi_giss.nc
     gmi_giss4x5_030520.nc
     gmit4_init.nc
```

More data files are available on `dirac` in */u1/gmidata/input_data/met_field/* and */u1/gmidata/input_data/misc/* directories under the corresponding subdirectory names.

3.3 Output Files

ASCII output and binary output files in netCDF data format are produced from GMI runs. The contents of the output files and the number of different output files can be controlled by using appropriate namelist parameters. To gain an understanding on how these parameters are set, read the *README.namelist* file.
For instance

```
by setting pr_flux = T, a file ..flux.nc is produced.
by setting pr_const=T, a file ..const.nc is produced.
setting pr_dry_depos = T, will print the dry deposition quantities.
```

Some sample output files are:

```
gmit4_giss_apr.qj.nc
gmit4_giss_apr_2x.const.nc
gmit4_giss_apr_2x.flux.nc
gmit4_giss_apr_2x.rst.nc
gmit4_giss_apr_2x.tend.nc
```

4 Specific Runs

As for testing the model, we assume that the model will be run from the directory */scratch/usrid* on `daley`, */scr/usrid* on `halem` and */mnt/pvfs1/usrid* on `thunderhead`. Go to the directory in order to copy the appropriate input namelist.

Suppose that you want to carry out a CO run. The namelist file needs for such a run is *dao2co2_dao29_10.in*.

```
%cp $GEMHOME/gem/actm/gmimod/Other/test/par/infiles/dao2co2_dao29_10.in .
```

Edit the file *dao2co2_dao29_10.in* and provide the location of the *gmi_data/* directory (where the Met Fields reside) by setting

```
For daley and thunderhead
      gmi_data_dir           = '/home/usrid/MYGMI/gmi_gsfc/gmi_data'
For halem
      gmi_data_dir           = '/u1/usrid/MYGMI/gmi_gsfc/gmi_data'
```

The met input file needed for this run is *DAO_GS_4x5x29_951101.nc* that already resides in the *gmi_data/* directory.

For output files, the following setting

```
pr_diag    = F,
pr_ascii   = T,
pr_ascii3  = F,
pr_netcdf  = T
```

will produce

```
dao2co2_dao29_10.asc
dao2co2_dao29_10.const.nc
```

Remark 5 *Three namelist variables are important to choose the number of processors. They are NP_actm, NPI_actm and NPJ_actm, and they satisfy the relation $NP_actm = NPI_actm \times NPJ_actm$. To execute the code, the number of processors $Ncpus$ is given by $Ncpus = NP_actm + 1$.*

Here are examples of batch script files (named *gmiCOrun.job*) for `daley`, `halem` and `thunderhead` respectively. The run requires 16 processors.

gmiCOrun.job on daley

```
#!/bin/csh -fx
#PBS -N gmi_COrun
#PBS -l ncpus=16
#PBS -l walltime=00:05:00
##PBS -l mem=2gb
#PBS -A k3002
#PBS -S /bin/csh
#PBS -V
#PBS -e error.file
#PBS -o output.file
#
cd /scratch/usrid
mpirun -np 16 $GEMHOME/bin/gem -d dao2co2_dao29_10.in
```

gmiCOrun.job on halem

```
#!/bin/csh
#BSUB -P a930b
#BSUB -J gmiCOrun
#BSUB -n 16
#BSUB -W 00:20
#BSUB -q general_lng
#BSUB -o /scr/usrid/gmiout.%J
#BSUB -e /scr/usrid/gmierr.%J
#
cd /scr/usrid
prun -n 16 $GEMHOME/bin/gem -d dao2co2_dao29_10.in
exit
```

gmiCOrun.job on thunderhead

```
### Number of nodes, processes and minutes requested
Nno=8
Npr=16
Nmi=60
ltmbegin -n $Nno -m $Nmi
#
WORK_DIR=/home/usrid/MYGMI/gmi_gsfc/gem
RUN_DIR=/mnt/pvfs1/pvfs1/usrid
cd $RUN_DIR
cp $WORK_DIR/bin/gem /home/usrid/rhome/MYGMI/gmi_gsfc/gem/bin
### Submit the job
time ltmpi -n $Nno -t $Npr=16 $WORK_DIR/bin/gem -d dao2co2_dao29_10.in
### End the session
ltmend
```

5 Summary of the Necessary Steps

In this section, we give the list of steps needed to obtain, install and run the GMI code on any platform.

1. Obtain the code (*gmi_gsfc* release) from the cvs repository.
2. Move to the GMI working directory (*gmi_gsfc/*).
3. Edit the file *cshrc.ggmi* to update the variables `GEMHOME`, `GMI_DATA` and `CHEMCASE`.
4. Copy the files *cshrc.ggmi* and *login.ggmi* to *.cshrc.ggmi* and *.login.ggmi* in your home directory.
5. Go to the directory *include/* to edit the files *gem_sys_options.h* and *gem_config.h* to select the architecture and to update the compilation options respectively.
6. Go to your home directory to edit and source the files *.cshrc* and .login.
7. Type `cd gem` and compile the code by typing `mkmf` then `make`.
8. Select a test case and in your input namelist file, update the variable `gmi_data_dir` providing the location of the input data.
9. Write a job script file and submit the job to run the executable.

References

[1] D.B. Considine, D.B., A.R. Douglass, P.S. Connell, D.E. Kinnison, and D.A. Rotman, A polar stratospheric cloud parameterization for the global modeling initiative three-dimensional model and its response to stratospheric aircraft, *J. Geophy. Res.*, **105**(D3), p. 3955-3973 (2000).

[2] A.R. Douglass, M.J. Prather, T.M. Hall, S.E. Strahan, P.J. Rasch, L.C. Sparling, L. Coy, and J.M. Rodriguez, Choosing meteorological input for the global modeling initiative assessment of high-speed aircraft, *J. Geophys. Res.*, **104**(D22), p. 27545-27564 (1999).

[3] D.E. Kinnison, et al., The Global Modeling Initiative assessment model: Application to high-speed civil transport perturbation, *J. Geophys. Res.*, **106**(D2), p. 1693-1711 (2001).

[4] D.A. Rotman, et al, Global modeling initiative assessment model: model description, integration, and testing of the transport shell, *J. Geo. Research*, **106**(D2), p. 1669-1691 (2001).