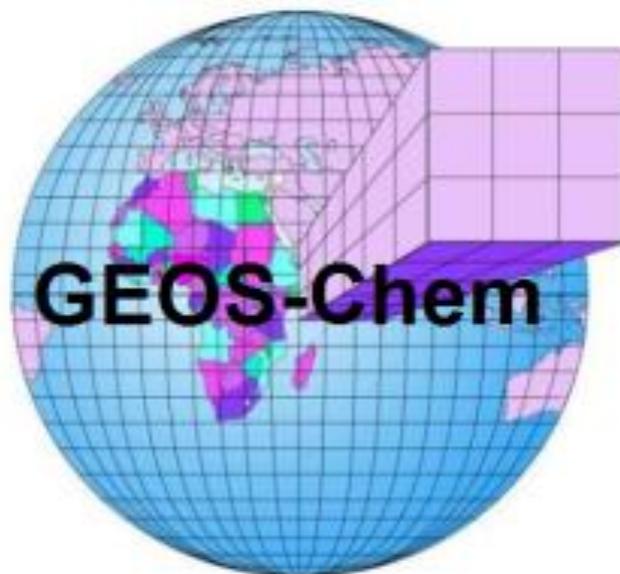


GEOS-Chem Model Clinic

Part 1: Beginner Topics

GEOS-Chem Support Team
geos-chem-support@as.harvard.edu



The 8th International GEOS-Chem Meeting
01 May 2017

Overview

- 1) Getting started with GEOS-Chem
- 2) Obtaining GEOS-Chem source code and data
- 3) Setting up GEOS-Chem simulations
- 4) Debugging Tips
- 5) Good coding practice

Getting Started with GEOS-Chem

GEOS-Chem Documentation

- GEOS-Chem Website:
<http://www.geos-chem.org>
- GEOS-Chem Wiki:
<http://wiki.geos-chem.org>
- GEOS-Chem Users' Guide:
<http://manual.geos-chem.org>
- GAMAP Users' Guide:
<http://acmg.seas.harvard.edu/gamap/doc/>

Joining GEOS-Chem Working Groups

All GEOS-Chem users are invited to join the Working Groups that corresponds to your own particular areas of research.

You are also encouraged to sign up for the relevant [WG email list\(s\)](#) and add your project to the group's page on the GEOS-Chem wiki:

Adjoint Model and Data Assimilation	geos-chem-adjoint@g.harvard.edu
Aerosols	geos-chem-aerosols@g.harvard.edu
Carbon Cycle	geos-chem-carbon@g.harvard.edu
Chemistry-Climate	geos-chem-climate@g.harvard.edu
GCHP	geos-chem-hp@g.harvard.edu
Hg and POPs	geos-chem-hg-pop@g.harvard.edu
Organics	geos-chem-organics@g.harvard.edu
Oxidants and Chemistry	geos-chem-oxidants@g.harvard.edu
Nested Model	geos-chem-regional@g.harvard.edu
Sources and Surface Uptake	geos-chem-emissions@g.harvard.edu
Transport	geos-chem-transport@g.harvard.edu

Getting Started

- Review the [GEOS-Chem basics](#) page
- Make sure you meet the [minimum system requirements](#)
- Requirements for running GEOS-Chem requirements include:
 - [A computer with a Unix distribution](#)
 - [GNU Make](#)
 - [Git source code management system](#)
 - [A Fortran compiler](#): ifort, pgfortran, or gfortran (new in v11-01!)
 - [A netCDF library installation](#)
 - [A GEOS-Chem source code directory](#)
 - [The GEOS-Chem Unit Tester](#) for creating run directories
 - [The GEOS-Chem shared data directories](#)
 - [Sample restart files for GEOS-Chem](#)
 - [A visualization package](#)
- This presentation includes links for accessing additional information

Git Version Control

- Git version control software is a distributed source code management system
- Offers many improvements over other source code management software (e.g. CVS, Subversion)
 - Allows you to keep an identical copy of the repository on your own system
 - Includes the option to create branches for different streams of development
 - Lets you easily apply and merge code from several developers
 - Contains easy-to-use graphical tools
- Git must be installed on your system
 - Git comes pre-packaged with several Unix OS builds
 - Otherwise, the Git source code can be obtained from the [Git website](#)
- GEOS-Chem source code, GEOS-Chem Unit Tester, and GAMAP visualization software are distributed with Git
 - GEOS-Chem repositories are stored on Bitbucket: <https://bitbucket.org/gcst/>

Git Graphical Tools

`gitk`

- This command opens the GitK window where you can browse the complete revision history
- For each commit, you can see the changes made to each file

Git Graphical Tools

gitk

The screenshot displays the gitk graphical interface for a repository named 'Code.v11-01'. The top window shows a commit history graph with branches 'v11-01-public-release', 'Dev', 'master', and 'remotes/origin/master'. The current commit is 'v11-01-precision=4' with the message 'Avoid array out-of-bounds error when reading jv_spec_mie.dat with brown c'. The commit details pane shows the author 'Melissa Sulprizio' and the commit message 'Update README.md for v11-01 public release'. The diff view shows the changes to 'README.md', including the addition of a new section for the 'v11-01-precision=4' release.

gitk: Code.v11-01

File Edit View Help

SHA1 ID: 7ab48bda0dfc186047aa49c54d7fd9e32acfa2ca Row 1 / 4551

Find commit containing: Exact All fields

Search

◆ Diff ◆ Old version ◆ New version Lines of context: 3 Ignore space change Line diff

◆ Patch ◆ Tree

Comments
README.md

Author: Melissa Sulprizio <mpayer@seas.harvard.edu> 2017-02-01 10:48:24
Committer: Melissa Sulprizio <mpayer@seas.harvard.edu> 2017-02-01 10:48:24
Tags: v11-01-public-release
Parent: bdb7d19fd01c4fbd9408c2af49a88d2d6a3c5534 (Minor fix for ND68 entries in tra
Branches: Dev, master, remotes/origin/master
Follows: v11-01-precision=4
Precedes:

Update README.md for v11-01 public release

Signed-off-by: Melissa Sulprizio <mpayer@seas.harvard.edu>

----- README.md -----
index 457753a..1c978d2 100644
@@ -1,10 +1,11 @@
README -- Describes contents of GROS-Chem repository
2017-02-01

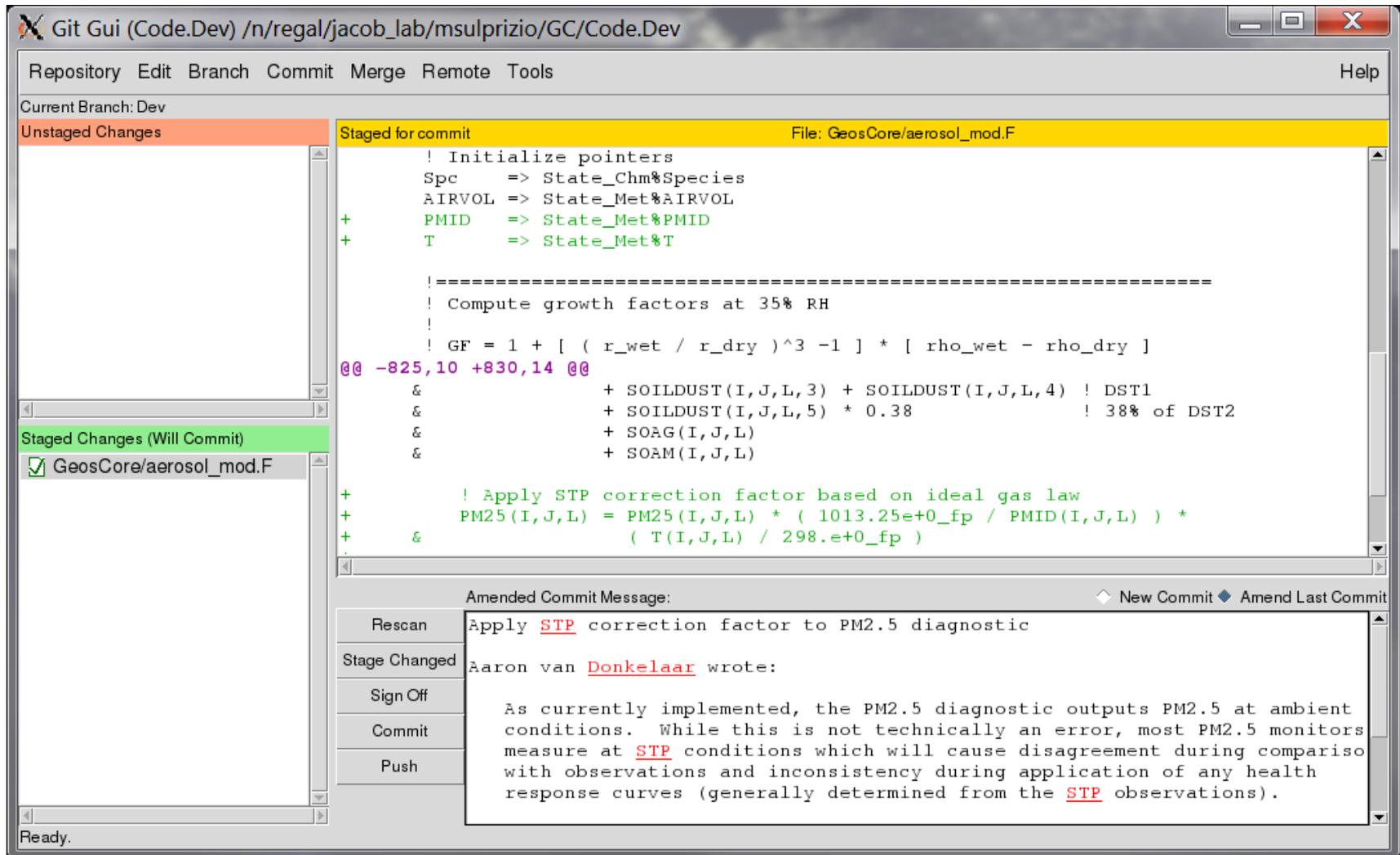
Git Graphical Tools

`git gui`

- This command opens the Git GUI, a graphical user interface that can be used for most Git commands
 - Create new branches of development
 - Commit changes to the repository
 - Merge branches back into the “master” branch

Git Graphical Tools

git gui



The screenshot shows the Git GUI application window titled "Git Gui (Code.Dev) /n/regal/jacob_lab/msulprizio/GC/Code.Dev". The interface includes a menu bar with "Repository", "Edit", "Branch", "Commit", "Merge", "Remote", "Tools", and "Help". The "Current Branch: Dev" is displayed at the top left. The main window is divided into several sections:

- Unstaged Changes:** An empty list.
- Staged Changes (Will Commit):** A list containing "GeosCore/aerosol_mod.F" with a checked checkbox.
- Staged for commit:** A yellow header bar with the file name "File: GeosCore/aerosol_mod.F". Below it, the code content is displayed, showing changes to initialize pointers and compute growth factors.
- Amended Commit Message:** A text area containing the commit message: "Apply STP correction factor to PM2.5 diagnostic".
- Buttons:** "Rescan", "Stage Changed", "Sign Off", "Commit", and "Push" are visible on the left side. "New Commit" and "Amend Last Commit" are visible on the right side.

```
! Initialize pointers
SpC  => State_Chm%Species
AIRVOL => State_Met%AIRVOL
+   PMID  => State_Met%PMID
+   T      => State_Met%T

! =====
! Compute growth factors at 35% RH
!
! GF = 1 + [ ( r_wet / r_dry )^3 -1 ] * [ rho_wet - rho_dry ]
@@ -825,10 +830,14 @@
&         + SOILDUST(I,J,L,3) + SOILDUST(I,J,L,4) ! DST1
&         + SOILDUST(I,J,L,5) * 0.38             ! 38% of DST2
&         + SOAG(I,J,L)
&         + SOAM(I,J,L)

+         ! Apply STP correction factor based on ideal gas law
+         PM25(I,J,L) = PM25(I,J,L) * ( 1013.25e+0_fp / PMID(I,J,L) ) *
+         &         ( T(I,J,L) / 298.e+0_fp )
```

Amended Commit Message: ◆ New Commit ◆ Amend Last Commit

Rescan Apply STP correction factor to PM2.5 diagnostic

Stage Changed Aaron van Donkelaar wrote:

Sign Off

Commit

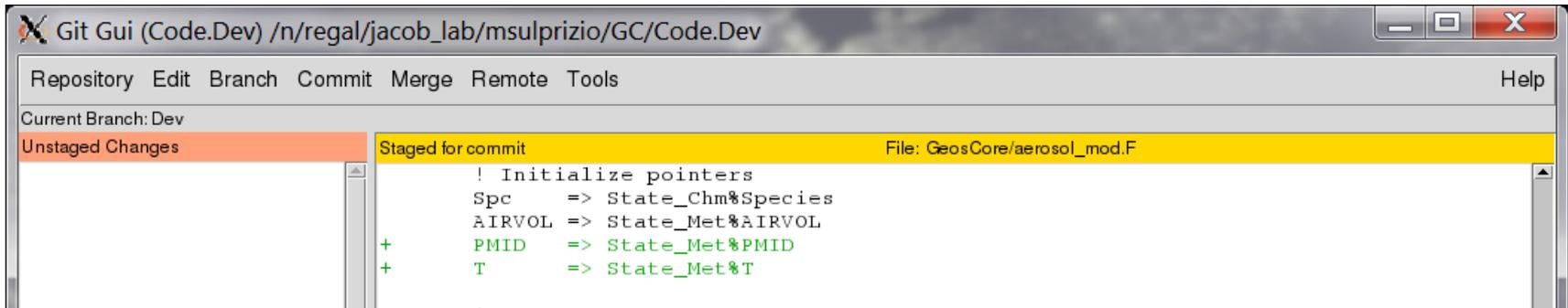
Push

As currently implemented, the PM2.5 diagnostic outputs PM2.5 at ambient conditions. While this is not technically an error, most PM2.5 monitors measure at STP conditions which will cause disagreement during comparison with observations and inconsistency during application of any health response curves (generally determined from the STP observations).

Ready.

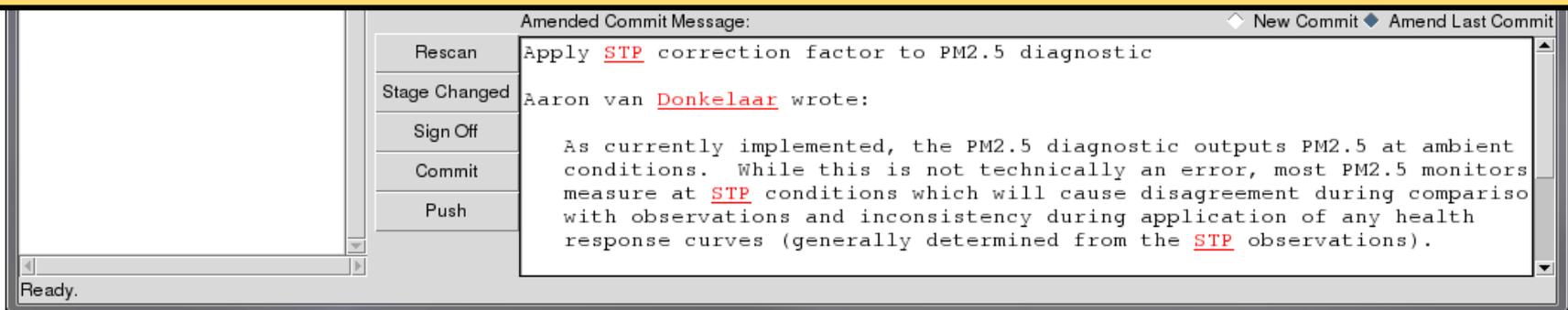
Git Graphical Tools

git gui



To simplify things, we recommend using the Gitk and Git GUI windows rather than typing commands at the Unix prompt.

For more information see [Using Git with GEOS-Chem](#) and [Writing good commit messages](#)



Obtaining GEOS-Chem Source Code and Data

Downloading Git Repositories

- GEOS-Chem repositories are now stored on Bitbucket
- Download the GEOS-Chem source code using Git:

```
git clone https://bitbucket.org/gcst/geos-chem Code.v11-01
```

- Install the netCDF libraries:
 - Use `module load netcdf` to see if netCDF libraries are on your system
 - Otherwise use GEOS-Chem-Libraries package

```
git clone https://bitbucket.org/gcst/geos-chem-libraries
```

- Link to the libraries from GEOS-Chem in your setup file (`.bashrc`, `.cshrc`).

- Download the GEOS-Chem Unit Tester

```
git clone https://bitbucket.org/gcst/geos-chem-unittest UT
```

Downloading Shared Data Directories

- The GEOS-Chem shared data directories contain input data read in during a simulation, such as:
 - [Meteorological data](#)
 - [Emissions inventories](#)
 - [Other atmospheric data](#) (e.g. prod/loss rates, oxidant fields)
 - [Sample restart files](#)
- You may download the GEOS-Chem shared data directories from one of the following archives using the `wget` utility
 1. Harvard archive (<ftp.as.harvard.edu>)
 2. Dalhousie archive (<rain.ucis.dal.ca>)
 3. Compute Canada archive (<http://geoschemdata.computecanada.ca>)

For example:

```
wget -r -nH ftp://ftp.as.harvard.edu/gcgrid/geos-chem/data/DIRNAME/
```

Downloading Shared Data Directories

- The GEOS-Chem shared data directories contain input data read in during a simulation, such as:
 - [Meteorological data](#)
 - [Emissions inventories](#)
 - [Other atmospheric data](#) (e.g. prod/loss rates, oxidant fields)
 - [Sample restart files](#)
- You may download the GEOS-Chem shared data directories from one of the following archives using the `wget` utility
 1. Harvard archive (<ftp.as.harvard.edu>)  **recommended for emissions data**
 2. Dalhousie archive (<rain.ucis.dal.ca>)  **recommended for met fields**
 3. Compute Canada archive (<http://geoschemdata.computecanada.ca>)  **for 0.5 x 0.25 global MERRA-2 met fields**

For example:

```
wget -r -nH ftp://ftp.as.harvard.edu/gcgrid/geos-chem/data/DIRNAME/
```

Meteorology Fields

Met field	Resolution	Availability
GEOS-FP	Global: 4° x 5°, 2° x 2.5° Nested: 0.25° x 0.3125°, 0.5° x 0.625°	2012/04/01 – present
MERRA-2	Global: 4° x 5°, 2° x 2.5°, 0.5° x 0.625° Nested: 0.5° x 0.625°	1980/01/01 – present
GEOS-5	Global: 4° x 5°, 2° x 2.5° Nested: 0.5° x 0.666°	2003/12/01 – 2013/06/30
MERRA	Global: 4° x 5°, 2° x 2.5°	1979/08/01 – 2015/02/28

- Nested grids are available for:
 - China: GEOS-5, GEOS-FP
 - Extended Asia: GEOS-FP, MERRA-2
 - Europe: GEOS-5, GEOS-FP, MERRA-2
 - North America GEOS-5, GEOS-FP, MERRA-2

Setting up GEOS-Chem Simulations

Creating GEOS-Chem Run Directories

- GEOS-Chem simulations are performed in a run directory
- Use a different run directory for each combination of met data, grid, and simulation type (e.g. `geosfp_4x5_fullchem`)
- Run directories for GEOS-Chem v11-01 are created with the [GEOS-Chem Unit Tester](#)
 1. Navigate to the `perl` subdirectory
 2. Edit `CopyRunDirs.input` to select the run directories that you wish to create
 3. Execute the `gcCopyRunDirs` perl script to create the run directories:

```
./gcCopyRunDirs CopyRunDirs.input
```
- Detailed instructions can be found on the [Creating GEOS-Chem run directories](#) wiki page

Setting Run Options

- Configurable files in a run directory include:
 - **Makefile**: compiles and runs GEOS-Chem
 - **input.geos**: sets run options
 - **HEMCO_Config.rc**: sets emission options
- Check the following settings in **Makefile**
 - **CODE_DIR**: GEOS-Chem source code directory path
 - **LOG_DIR**: output log path
 - **VERSION**: prefix for output log filename
- Check the simulation settings in **input.geos**
 - Start and end dates
 - Data directory file paths
 - Options for transport, convection, deposition, chemistry, etc.
 - Options for diagnostics and timeseries output

Setting Emission Options

- Check the emissions settings in `HEMCO_Config.rc`

```
#####  
### BEGIN SECTION SETTINGS  
#####
```

```
ROOT:                /LOCAL_PATH/gcgrid/data/ExtData/HEMCO  
Logfile:             HEMCO.log  
DiagnFile:           HEMCO_Diagnostics.rc  
DiagnPrefix:         HEMCO_Diagnostics  
DiagnFreq:           End  
Wildcard:            *  
Separator:           /  
Unit tolerance:     1  
Negative values:    0  
Verbose:             0  
Warnings:           1
```

- **ROOT:** path to the HEMCO data directories
- **Logfile:** name of file where errors, warning, and debug output is sent (default = `HEMCO.log`)
- **DiagnFile, DiagnPrefix, DiagnFreq:** HEMCO emission diagnostics settings
 - Frequency options are `Hourly`, `Daily`, `Monthly`, `Annual`, `End`, or `Manual`
 - The emission diagnostics will be saved out to either the GEOS-Chem bpch file or the HEMCO netCDF diagnostic file, but not both
- **Verbose, Warnings:** control the amount of output sent to `HEMCO.log` (0=least, 3=most)

Setting Emission Options

- Use **true/false** to specify emission inventories

```
# ExtNr ExtName                on/off Species
0      Base                    : on    *
      --> HEMCO_RESTART         :      true
      --> AEIC                  :      true
      --> BIOFUEL                :      true
      --> BOND                   :      true
      --> BRAVO                  :      true
      --> CAC                    :      true
```

- Use **on/off** to specify emission extensions (e.g. MEGAN, GFED)

```
108    MEGAN                    : on    ISOP/ACET/PRPE/C2H4/ALD2/EOH
      --> Isoprene scaling       :      1.0
      --> CO2 inhibition         :      true
      --> CO2 conc (ppmv)       :      390.0
109    MEGAN_Mono               : on    CO/OCPI/MONX
110    MEGAN_SOA                : on    MTPA/MTPO/LIMO/SESQ
```

Compiling GEOS-Chem

- You can compile and run your GEOS-Chem simulation from your run directory
 - We recommend executing make commands in the run directory instead of the top-level code directory to automatically select proper options
- Some important compiling commands:
 - `make help`: shows help screen with all possible make options
 - `make -j4`: compiles 4 files simultaneously (optional; use # of CPUs available)
 - `make -j4 all`: runs a GC unit test (`make sp`, `make mp`, `make check`)
 - `make -j4 mpbuild`: compiles GEOS-Chem with parallelization on
 - `make -j4 mp`: compiles & runs GEOS-Chem with parallelization on
 - `make -j4 mp BOUNDS=yes`: compiles & runs w/ array-out-of-bounds checks
 - `make -j4 mp FPE=yes`: compiles & runs w/ floating-point exception checks
 - `make fileclean`: removes all output files in the run directory
 - `make realclean`: removes all compiled files in the code directory
 - `make superclean`: runs `make fileclean realclean`

Submitting Runs

- We recommend testing your simulation with a short (1-day) run before submitting a long-term simulation
- GEOS-Chem runs can be submitted interactively or using a batch queue system
 - If your system supports interactive sessions, you can run GEOS-Chem from the Unix prompt (e.g. `./geos >& gc.log`)
 - If your system uses a batch queue system (e.g. SLURM, SGE, PBS), you can create a run script and submit it to the scheduler
 - A sample run script (`v11-01.run`) is included in the `4x5_standard` run directory in the GEOS-Chem Unit Tester
- Sample [restart files](#) are provided for initializing simulations
 - GEOS-Chem restart files contain species concentrations (**Required**)
 - HEMCO restart files contain values needed for HEMCO extensions (**Optional**)
 - The sample GEOS-Chem restart files do not reflect the actual atmospheric state and should only be used to "spin up" the model

Debugging Tips

Debugging Tips

Here are some tips to isolate the source of errors:

- 1) Look at the log file(s)
- 2) Make sure you didn't max out the allotted time or memory
- 3) Check if someone else has already reported the bug
- 4) Rerun with debug flags and verbose options turned on
- 5) Identify whether the error happens consistently
- 6) Use a debugger to locate the source of the error
- 7) Isolate the problem to a particular area of the code
- 8) Revert to a clean copy of the code
- 9) Focus on areas of the code that you modified
- 10) Check for math errors (divide by zero, log of negative, etc.)
- 11) When in doubt, print it out!

Debugging GEOS-Chem

- Check both the GEOS-Chem log file and the HEMCO log file
 - Turn on `ND70` in `input.geos` to print debug output to the log file
 - Set `Verbose` and `Warnings` to 3 in `HEMCO_Config.rc`
- Compile with debug options turned on
 - `BOUNDS=yes` turns on array out-of-bounds error checking
 - `TRACEBACK=yes` returns a list of routines that were called when the error occurred (on by default in v11-01 and later)
 - `FPE=yes` checks for floating-point exceptions (div-by-zero, NaN, etc.)
 - `DEBUG=yes` compiles GEOS-Chem for use in a debugger
- Isolate the error to a particular operation (e.g. transport, chemistry, wet deposition) by turning on one at a time
- If stuck, send the GCST an email and include the log files

Please report any bugs in the standard code to the GCST and post on the GEOS-Chem wiki for other users to see!

Fortran Debuggers

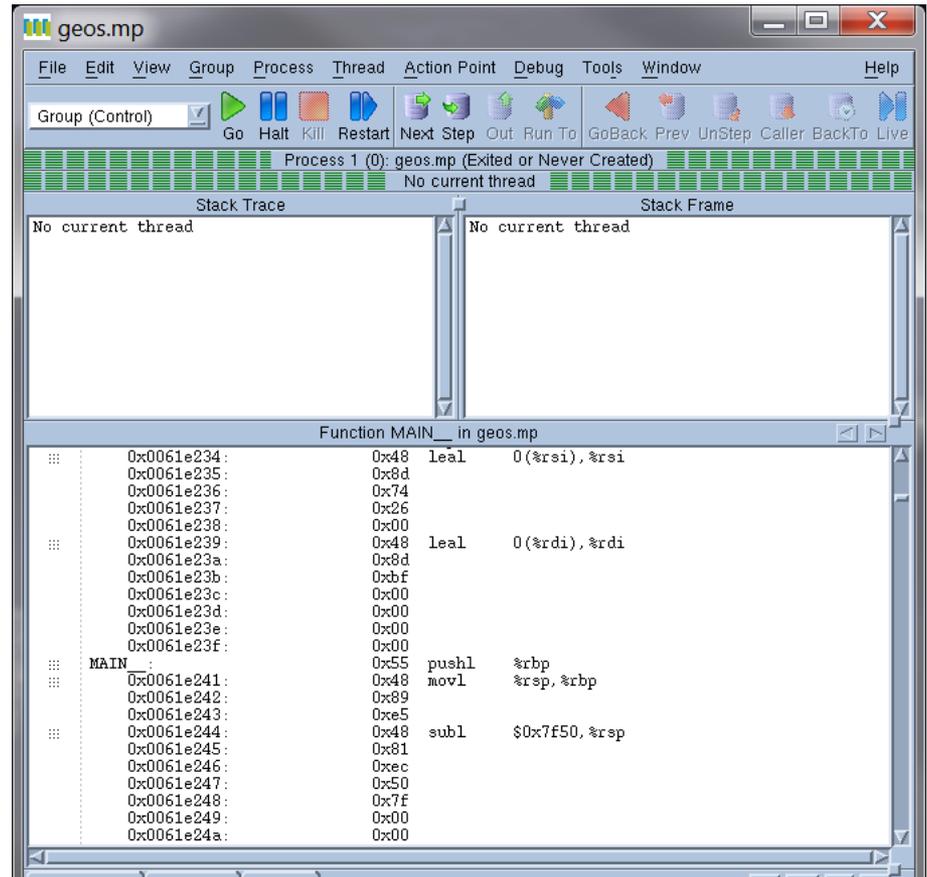
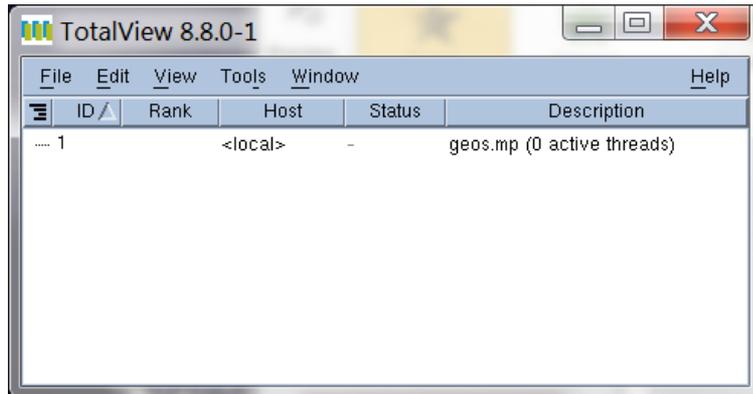
- Using a debugger can save you a lot of time and hassle
- Debuggers allow you to:
 - Examine data when a program stops
 - Navigate the stack when a program stops
 - Set break points
- Common debuggers include TotalView, IDB, GDB, DBX
- For GEOS-Chem, add **DEBUG=yes** to the make command

```
make -j4 mpbuild DEBUG=yes      # With      parallelization
make -j4 spbuild DEBUG=yes      # Without  parallelization
```

- **DEBUG=yes** evokes
 - g to show source code lines in debugger instead of hexadecimal notation
 - O0 to switch off compiler optimization, which will make the code execute exactly as it is written

TotalView

- TotalView is a GUI-based source code and memory debugging tool for parallel applications written in C, C++, and Fortran
- To use type `totalview geos.mp`



Good Coding Practice

Good Coding Practice

- Use copious comments
 - Comments will make sure that other users will be able to understand the modifications that you have added
 - Writing comments before actually writing the code may help in your planning
 - Well documented code is as important as properly working code
- Tips for adding comments
 - Flag your updates with your initials and the date, e.g. (`mps , 2/23/17`)
 - Separate sections of the code with one or more divider comments
 - When citing papers, make sure to include the complete citation at the top of the routine and use shorthand to reference that paper within the code
 - Limit line length to avoid wrapping the text on the page

Good Coding Practice

- Use a text editor with syntax highlighting
- Use ProTeX headers
 - ProTeX headers are specially formatted comments at the top of each subroutine, function, and module that list arguments, modification history, and references
 - Documentation is automatically generated when you compile with `make doc`
- Follow these tips for writing effective code
 - Indent new blocks of code within loops and if statements
 - The exception to this is nested loops (e.g. looping over lat, lon, lev)
 - Use lots of white space to separate code
 - Use all capitals for Fortran reserve words
 - You may at your discretion mix capitals and lowercase in variable names
 - Structure loops efficiently
 - In Fortran loop over arrays using **N, L, J, I** order
 - Loops in the wrong order can dramatically slow down your code

Coding Tips

- Understand the problem and break it down into small steps
 - Write pseudocode to focus on the logic without being distracted by syntax
 - Syntax varies between languages, but the thought process is the same
- Look for relevant examples in preexisting code and modify
 - Don't start from scratch if you can help it
 - Use Google, Stack Overflow, fellow group members
- Compile/run your code often
 - Comment out sections or routine calls to locate problematic code
 - Use print statements to check variables, determine where code stops, etc.
 - Fix bugs before writing new code
- Use version control
 - Lets you track your files over time so you can revert to a previous working version if necessary

GCST office hours

- The GEOS-Chem Support Team will be available for questions and demonstrations during IGC8

When: 1:30 – 3:00 PM Mon thru Wed

Where: Maxwell-Dworkin lobby

- Or email us anytime at

geos-chem-support@as.harvard.edu