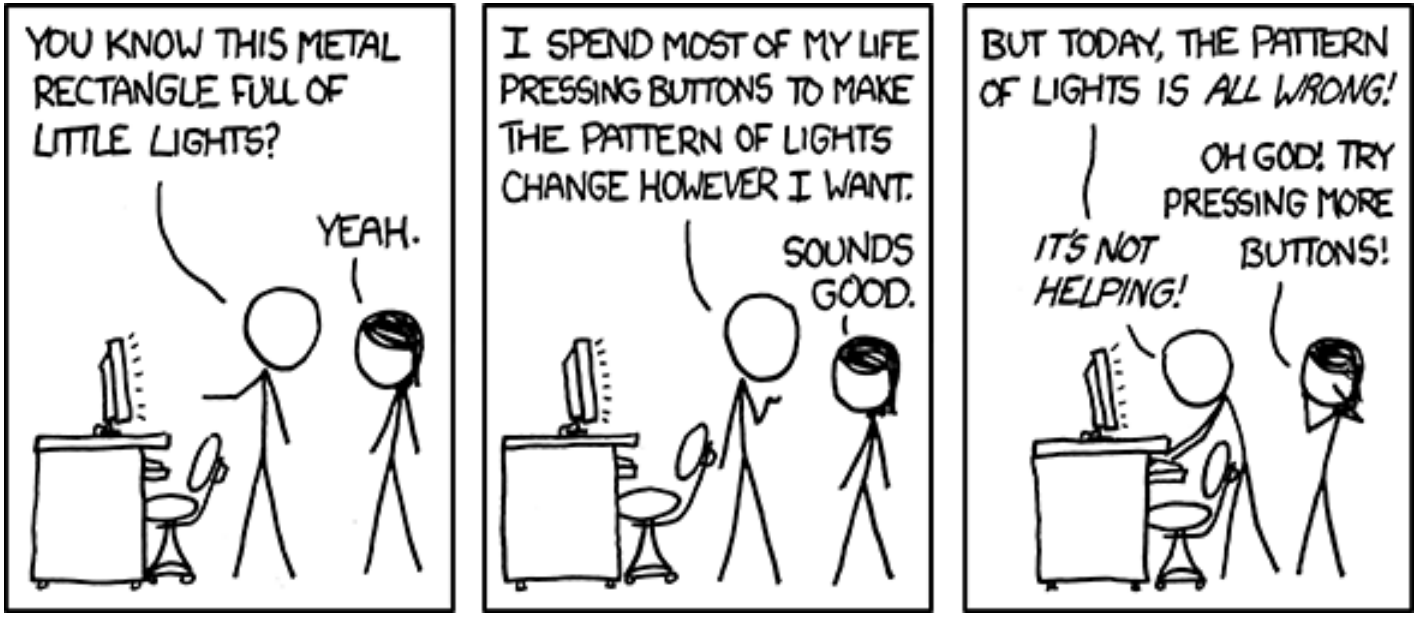


GEOS-Chem Model Clinic @ IGC8

Part 2: Advanced Topics

GEOS-Chem Unit Tester

HEMCO / Emissions

FlexChem / KPP chemistry solver

GEOS-Chem Species Database

GEOS-Chem Support Team

01 May 2017

Note

- This presentation contains a **LOT** of information on several important aspects of GEOS-Chem.
- We probably will not cover all of this material in a 30 minute Model Clinic.
- We invite you to download this PDF and refer to it (at a more leisurely pace) at a later time.

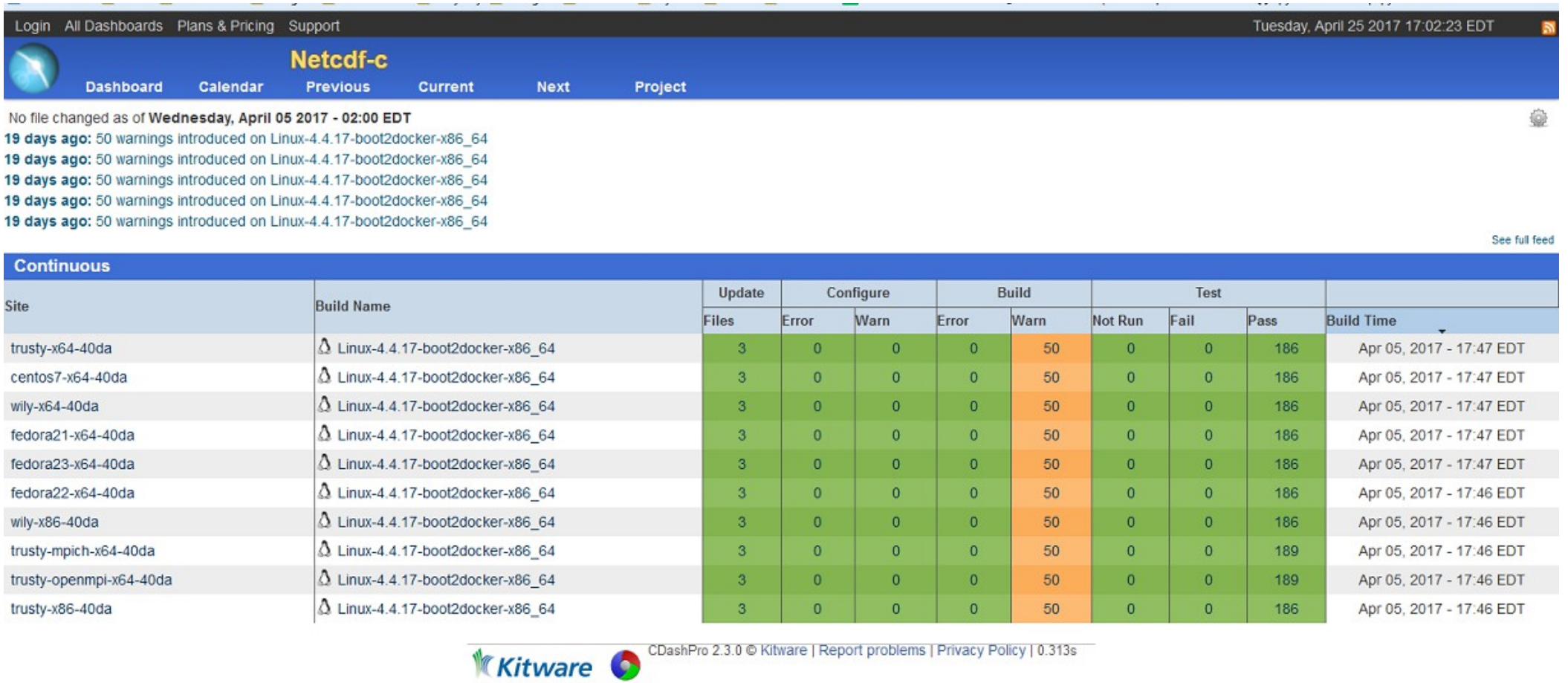
Setting up GEOS-Chem Unit Tests:
Debugging several simulations simultaneously

Introduced in v9-02

What is “Unit Testing”?

- Unit Tests are short simulations designed to reveal flaws in software.
- Unit Tests can be automated and run nightly, especially after a new feature has been added.
- Many software libraries that we use (e.g. ESMF, netCDF) run unit tests and post the results online.

Example: netCDF unit tests



Login All Dashboards Plans & Pricing Support — Tuesday, April 25 2017 17:02:23 EDT

Netcdf-c

Dashboard Calendar Previous Current Next Project

No file changed as of **Wednesday, April 05 2017 - 02:00 EDT**

19 days ago: 50 warnings introduced on Linux-4.4.17-boot2docker-x86_64
19 days ago: 50 warnings introduced on Linux-4.4.17-boot2docker-x86_64
19 days ago: 50 warnings introduced on Linux-4.4.17-boot2docker-x86_64
19 days ago: 50 warnings introduced on Linux-4.4.17-boot2docker-x86_64
19 days ago: 50 warnings introduced on Linux-4.4.17-boot2docker-x86_64

See full feed

Continuous

Site	Build Name	Update	Configure		Build		Test			
		Files	Error	Warn	Error	Warn	Not Run	Fail	Pass	Build Time
trusty-x64-40da	Linux-4.4.17-boot2docker-x86_64	3	0	0	0	50	0	0	186	Apr 05, 2017 - 17:47 EDT
centos7-x64-40da	Linux-4.4.17-boot2docker-x86_64	3	0	0	0	50	0	0	186	Apr 05, 2017 - 17:47 EDT
wily-x64-40da	Linux-4.4.17-boot2docker-x86_64	3	0	0	0	50	0	0	186	Apr 05, 2017 - 17:47 EDT
fedora21-x64-40da	Linux-4.4.17-boot2docker-x86_64	3	0	0	0	50	0	0	186	Apr 05, 2017 - 17:47 EDT
fedora23-x64-40da	Linux-4.4.17-boot2docker-x86_64	3	0	0	0	50	0	0	186	Apr 05, 2017 - 17:47 EDT
fedora22-x64-40da	Linux-4.4.17-boot2docker-x86_64	3	0	0	0	50	0	0	186	Apr 05, 2017 - 17:46 EDT
wily-x86-40da	Linux-4.4.17-boot2docker-x86_64	3	0	0	0	50	0	0	186	Apr 05, 2017 - 17:46 EDT
trusty-mpich-x64-40da	Linux-4.4.17-boot2docker-x86_64	3	0	0	0	50	0	0	189	Apr 05, 2017 - 17:46 EDT
trusty-openmpi-x64-40da	Linux-4.4.17-boot2docker-x86_64	3	0	0	0	50	0	0	189	Apr 05, 2017 - 17:46 EDT
trusty-x86-40da	Linux-4.4.17-boot2docker-x86_64	3	0	0	0	50	0	0	186	Apr 05, 2017 - 17:46 EDT

Kitware — CDashPro 2.3.0 © Kitware | Report problems | Privacy Policy | 0.313s

These are unit tests of the netCDF software library.
GREEN boxes indicate individual unit tests that have passed.
ORANGE boxes indicate the presence of (non-fatal) compilation warnings.

http://my.cdash.org/index.php?project=netcdf-c&date=2017-04-05

GEOS-Chem Unit Tester

- G-C Unit Tester is an external package
  - Downloadable via Git
  - `git clone https://bitbucket.org/gcst/geos-chem-unittest.git UT`
- The Unit Tester package has several subdirs:
  - `perl:` Contains scripts & input files
  - `runs:` Contains GEOS-Chem run directories
  - `logs:` where log file output will get sent
  - `jobs:` where job files will be created
  - `doc:` where documentation gets built

GEOS-Chem Unit Tester

- Each unit test validates a combination of:
  met field + horizontal grid + simulation type:
  - `geos5_2x25_CO2`
  - `merra_4x5_Hg`
  - `geosfp_4x5_standard`
  - `geos5_4x5_soa`
  - `merra2_05x0625_CH4_na` (denotes type of nested grid)
- See our wiki page:
  - http://wiki.geos-chem.org/GEOS-Chem_Unit_Tester

GEOS-Chem Unit Tester

- For each combination of met + grid + sim:
  - GC is run with strict debugging options, twice:
    - Once without OpenMP parallelization (aka "sp")
    - Once with OpenMP parallelization (aka "mp")

- The debug options will catch common errors:
  - `DEBUG=y BOUNDS=y FPE=y NO_ISO=y`

- Output files from "sp" and "mp" runs are checked. If not identical, there's a problem.

GC Unit Tester – Interactive demo!

- We will start a GC Unit Test job!
  - We will pick the unit tests that we wish to run by editing the `UnitTest.input` file in the `perl/` folder
  - Unit test is submitted with this command
    - `./gcUnitTest UnitTest.input`
  - See the *GEOS-Chem Unit Tester* wiki page

- While the job runs, we'll examine what types of bugs the Unit Tester scans for.

Check the *.results.log file

```
################################################################################
### VALIDATION OF GEOS-CHEM OUTPUT FILES
### In directory: geosfp_4x5_CH4
###
### File 1    : trac_avg.geosfp_4x5_CH4.201307010000.sp
### File 2    : trac_avg.geosfp_4x5_CH4.201307010000.mp
### Sizes     : IDENTICAL (8922424 and 8922424)
### Checksums : IDENTICAL (34317084 and 34317084)
### Diffs     : IDENTICAL
###
### File 1    : GEOSChem_restart.201307010020.nc.sp
### File 2    : GEOSChem_restart.201307010020.nc.mp
### Sizes     : IDENTICAL (651408 and 651408)
### Checksums : IDENTICAL (3831259697 and 3831259697)
### Diffs     : IDENTICAL
###
### File 1    : HEMCO_restart.201307010020.nc.sp
### File 2    : HEMCO_restart.201307010020.nc.mp
### Sizes     : IDENTICAL (28556 and 28556)
### Checksums : IDENTICAL (2542351414 and 2542351414)
### Diffs     : IDENTICAL
################################################################################
... etc for the other unit tests …
```

Debugging options

- Each unit test is designed to reveal:
  - Floating-point math exceptions
  - Array out-of-bounds errors
  - Inefficient subroutine calls
  - Parallelization errors

- Code optimization is disabled (**`-O0`**)
  - Optimization can sometimes mask bugs
  - This also facilitates use of a debugger (Totalview)

Floating-point math exceptions

- Denormal values
  - `-Infinity, +Infinity`
    - A value that is (smaller,larger) than the (min,max) value that can be represented in REAL*4 or REAL*8.
    - e.g. Result of a division by zero
  - `NaN`
    - “Not a number”: Result of an illegal computation.
    - SQRT(-1), LOG(0), LOG(-1), etc.
- Denormal values propagate through code
  - `X + NaN = NaN;  X * NaN = NaN; etc.`

Example #1. Pretend somebody slipped this into G-C:

```
! Force a div-by-zero error
A = 1.0
B = 0.0
PRINT*, 'Result:', A/B
```

If you didn't check for floating-point errors, you would get this result:

```
> Result: Infinity
```

But the GC Unit Tester (which uses debug option FPE=y) will stop with:

```
> forrtl: error (65): floating invalid
Image      PC                Routine Line     Source
geos       00000000004703B1  Unknown Unknown  Unknown
geos       000000000040325C  MAIN__  12       main.F
geos       000000000040319C  Unknown Unknown  geos
libc.so.6  00000037B2A21A05  Unknown Unknown  Unknown
geos       0000000000403099  Unknown Unknown  Unknown
```

Array out-of-bounds error

Example #2: Assume somebody slipped this into G-C:

```
      REAL*4 :: ARRAY(3)
      ARRAY(1) = 1.0
      ARRAY(2) = 2.0
      ARRAY(3) = 3.0
      ARRAY(4) = 4.0  !%%% This is outside of array bounds!
      PRINT*, 'Result: ', ARRAY(1:4)
```

Without checking for array out-of-bounds errors, you'd see this:

```
 > Result:  1.000000   2.000000   3.000000   4.000000
```

You may not notice anything wrong at first, but the run may die later on.

But the GC Unit Tester (using BOUNDS=y) would stop with:

```
> main.F(12): error #5561: Subscript #1 of the array ARRAY has value 4
which is greater than the upper bound of 3
      ARRAY(4) = 4.0
```

Finding inefficient subroutine calls

Example #3: Assume somebody slipped this into G-C:

```
! In the calling routine
REAL*4, ARRAY(IIPAR,JJPAR,LLPAR)
ARRAY = 100.0
CALL MY_ROUTINE( ARRAY(:,1,:) )  # Cannot pass this array slice
                                 # because dims are non-contiguous;
                                 # Compiler must create a temp array
                                 # to pass IIPAR x LLPAR slice,
                                 # which costs memory + CPU cycles.

! In the called subroutine
SUBROUTINE MY_ROUTINE( ARRAY )
REAL*4, INTENT(INOUT) :: ARRAY(IIPAR,LLPAR)
. . .
END SUBROUTINE MY_ROUTINE
```

But the GC Unit Tester (using DEBUG=y) will print:

```
> forrtl: warning (402): fort: (1): In call to MY_ROUTINE, an array
temporary was created for argument #1
```

Detecting parallelization errors

Example #4: Assume somebody slipped this into G-C:

```
!$OMP PARALLEL DO
!$OMP+DEFAULT( SHARED )
!$OMP+PRIVATE( I, J )    !%%% ERROR: P should be PRIVATE but isn't %%%
      DO J = 1, JJPAR
      DO I = 1, IIPAR
         P = A(I,J) * 2.0
         B(I,J) = P
      ENDDO
      ENDDO
!$OMP END PARALLEL DO
```

See our *Parallelizing GEOS-Chem* wiki page for more information about writing parallel DO loops!

But the GC Unit Tester compares the output of the "sp" and "mp" runs. If they are not identical, there is a parallelization error:

```
### File 1    : trac_avg.geosfp_4x5_fullchem.201307010020.sp
### File 2    : trac_avg.geosfp_4x5_fullchem.201307010020.mp
### Sizes     : IDENTICAL (63354688 and 63354688)
### Checksums : DIFFERENT (3169241717 and 2208286062)
### Diffs     : DIFFERENT
```

GEOS-Chem Unit Test Results

Version: v11-01-public-release
Date Submitted: 2017/01/27 17:48:02
Description: Tests GEOS-Chem v11-01-provisional-release

An HTML file will be created to graphically display the results of each unit test. This can be uploaded to a web page at specified intervals.

GCST typically refreshes the web page every 10 minutes in order to monitor the progress of ongoing unit tests.

4° x 5° Unit Tests	Standard	Trop chem	UCX	RR TMG	SOA	SOA-SVPOA	Acid Uptk	Marine POA	Rn-Pb-Be	Hg	TagHg	POPs	TagCO	TagO3	CH4	CO2	Aer-osol	TOMAS 15	TOMAS 40
MERRA-2 @ 4° x 5°																			
GEOS-FP @ 4° x 5°																			
MERRA @ 4° x 5°																			
GEOS-5 @ 4° x 5°																			
GEOS-4 @ 4° x 5°																			
GCAP @ 4° x 5°																			
2° x 2.5° Unit Tests	**Standard**	**Trop chem**	**UCX**	**RR TMG**	**SOA**	**SOA-SVPOA**	**Acid Uptk**	**Marine POA**	**Rn-Pb-Be**	**Hg**	**TagHg**	**POPs**	**TagCO**	**TagO3**	**CH4**	**CO2**	**Aer-osol**	**TOMAS 15**	**TOMAS 40**
MERRA-2 @ 2° x 2.5°																			
GEOS-FP @ 2° x 2.5°																			
GEOS-5 @ 2° x 2.5°																			

Nested-Grid Unit Tests	Trop chem	CH4	Hg
GEOS-5 NA @ 0.5° x 0.666°			
GEOS-5 CH @ 0.5° x 0.666°			
GEOS-FP NA @ 0.25° x 0.3125°			
GEOS-FP CH @ 0.25° x 0.3125°			
MERRA-2 NA @ 0.5° x 0.625°			
MERRA-2 AS @ 0.5° x 0.625°			

LEGEND

- The unit test was successful.
- Further investigation is necessary (e.g. The restart files were identical but the diagnostic output differed).
- The unit test failed.
- A unit test was not performed for this combination of met field, horizontal grid, and simulation type.

```
GEOS-Chem Unit Test                : RESULT
-----------------------------------------
geos5_05x0666_CH4_na               : GREEN
merra_2x25_Hg                      : GREEN
merra2_2x25_RnPbBe                 : GREEN
merra2_4x5_RnPbBe                  : GREEN
merra2_2x25_CO2                    : GREEN
geos4_4x5_Hg                       : GREEN
merra_4x5_tag03                    : GREEN
geos5_4x5_CH4                      : GREEN
geosfp_4x5_POPs                    : GREEN
geos5_2x25_CO2                     : GREEN
geosfp_2x25_CH4                    : GREEN
merra_4x5_Hg                       : GREEN
geos4_4x5_POPs                     : GREEN
geos5_2x25_RnPbBe                  : GREEN
geos5_2x25_tag03                   : GREEN
geosfp_4x5_tagCO                   : GREEN
geos5_4x5_POPs                     : GREEN
geos5_2x25_Hg                      : YELLOW
merra_4x5_tagCO                    : GREEN
geos5_4x5_RnPbBe                   : GREEN
merra2_4x5_Hg                      : GREEN
merra2_2x25_POPs                   : GREEN
merra2_4x5_tagCO                   : GREEN
```

A text file is also created to display the results of the unit tests.

This is useful if your computational cluster system will not let you upload data to a web server on the internet.

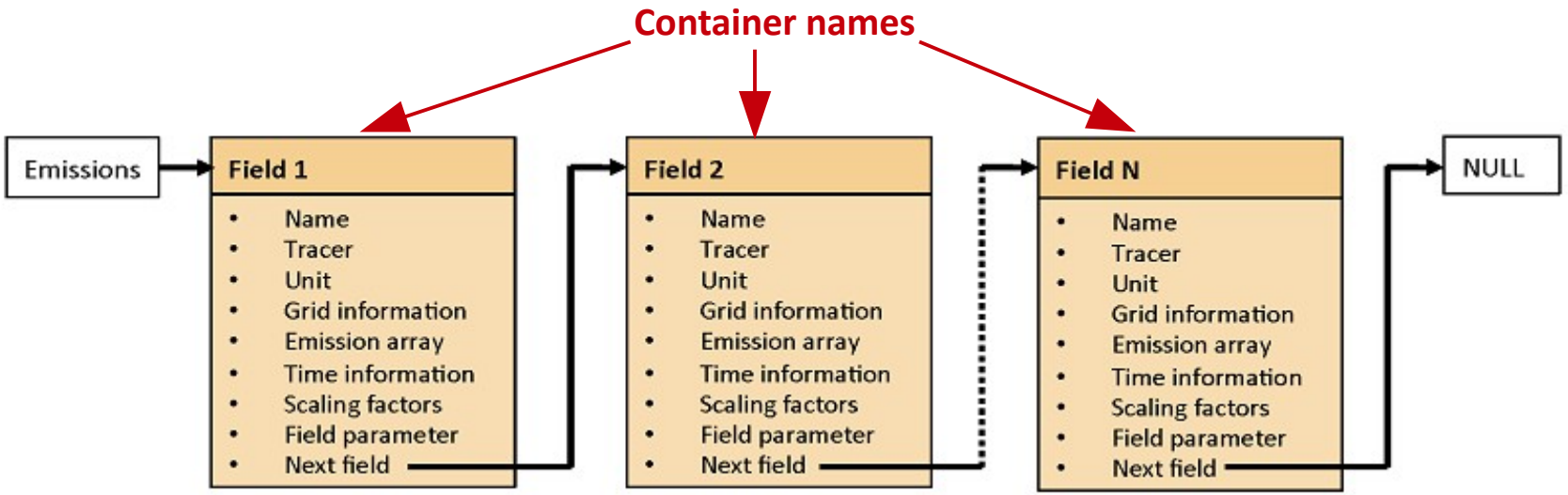
HEMCO:
A new way to compute emissions

Introduced in v10-01

HEMCO features

- User specifies emissions options in a config file
  - Base emissions: don't depend on met fields
  - Extensions: Pass met fields (P, T) to Fortran modules to compute resultant emissions
- Emissions stored in a “linked list” data structure
- Uses netCDF input data files
- See *The HEMCO User's Guide* for more examples
  - http://wiki.geos-chem.org/The_HEMCO_Users_Guide

The emissions linked list in HEMCO

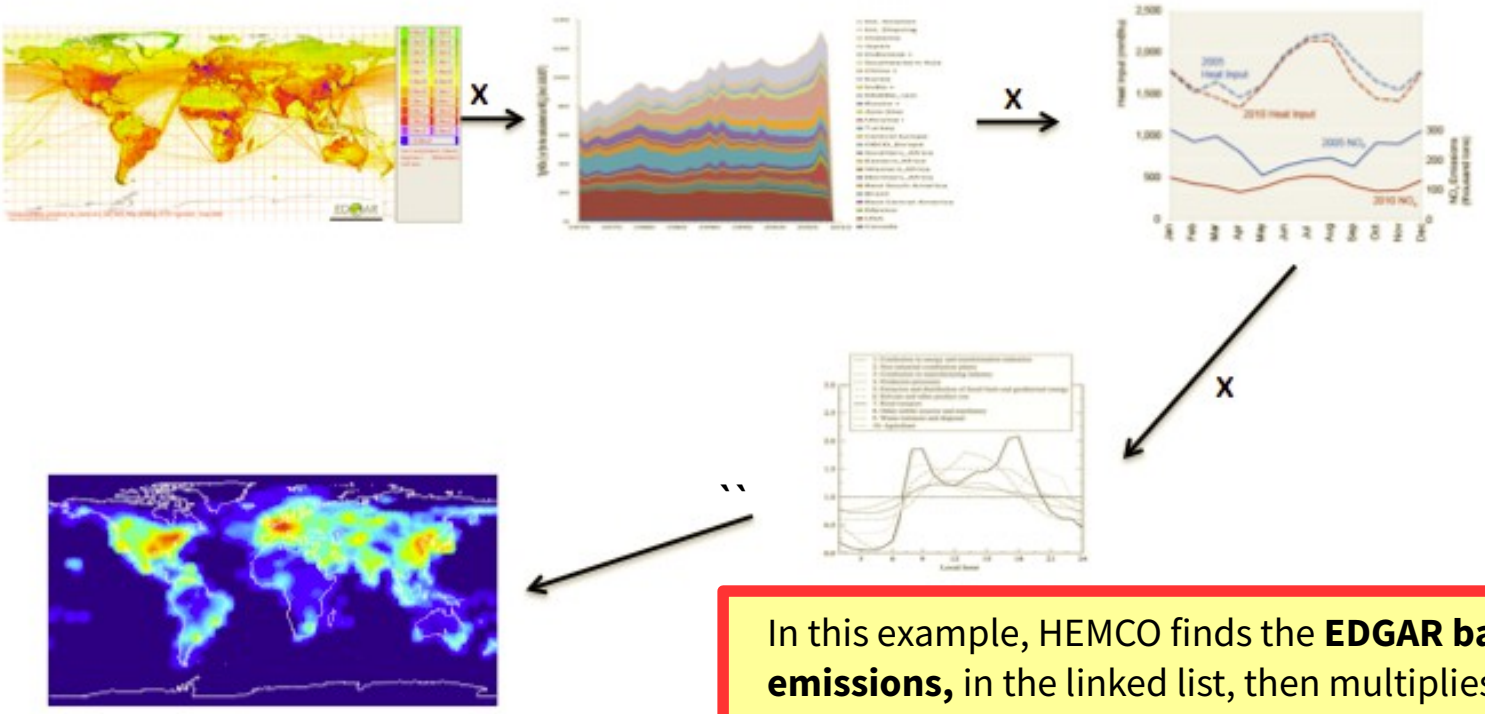


- List with all emission data
- Flexible length
- One variable ('Emissions') leads to all content
- Contains all information required to calculate the emissions

The heart and soul of HEMCO is the **linked list** structure. The linked list consists of multiple **containers**. Each **container** can store an **emissions inventory** (global or regional), a **non-emissions data set**, a **scale factor**, a **mask**, the **total computed emissions for a given species**, or a **diagnostic output**.

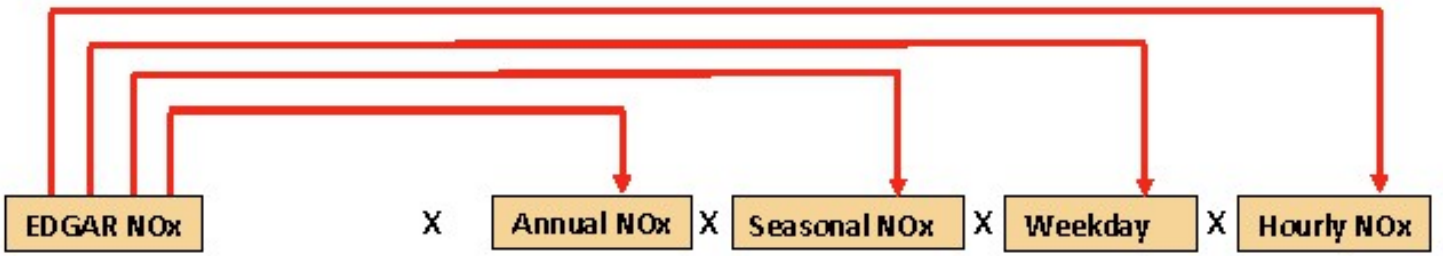
Courtesy of Christoph Keller

Applying the scale factors: Pointers!



In this example, HEMCO finds the **EDGAR base emissions,** in the linked list, then multiplies it only by the **scale factors** that are relevant. (You specify scale factors for each inventory in the configuration file.)

EDGAR NOx for given time step:



Courtesy of Christoph Keller

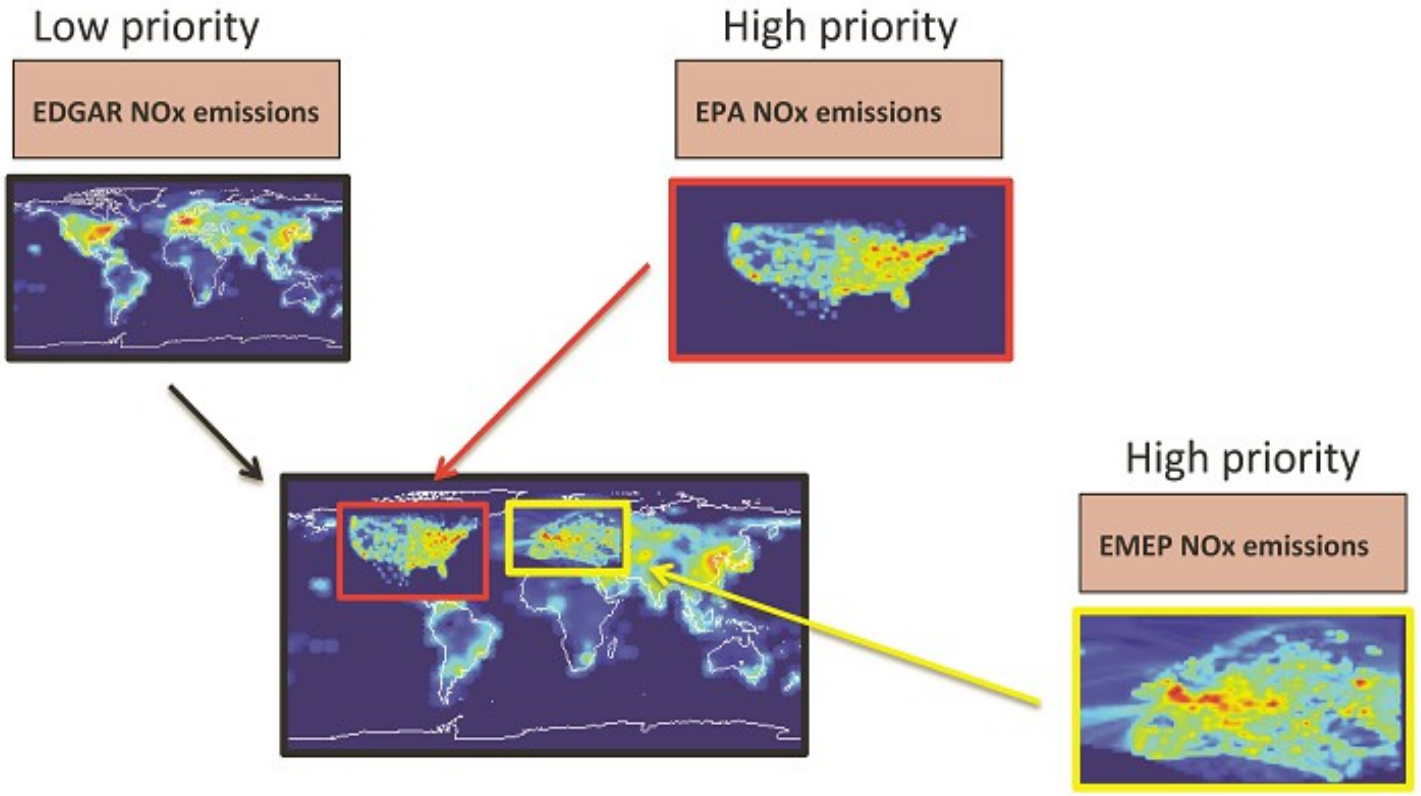
Field priorities

- Hierarchy of emissions defined through field priorities
- High-priority emissions overwrite low-priority emissions

In the HEMCO configuration file, you can specify the **priorities** for each species in each inventory.

Emissions of different ***categories*** are added together (e.g. anthro + biomass NO).

Emissions in the same category (e.g. anthro NO) may be assigned a ***hierarchy*** (higher values overwrite lower values).



Courtesy of Christoph Keller

The HEMCO Configuration file

1) Settings
2) Base Emission Switches
3) Base Emissions Data
4) Extensions Switches
5) Extensions Data
6) Scale Factors
7) Masks
8) Non-Emissions Data

HEMCO configuration file: *Settings*

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

ROOT:                           /n/holylfs/EXTERNAL_REPOS/GEOS-CHEM/gcgrid/data/ExtData/HEMCO
Logfile:                        HEMCO.log
DiagnFile:                      HEMCO_Diagnostics.rc
DiagnPrefix:                    HEMCO_diagnostics
DiagnFreq:                      End
Wildcard:                       *
Separator:                      /
Unit tolerance:                 1
Negative values:                0
Only unitless scale factors:    false
Verbose:                        0
Warnings:                       1

### END SECTION SETTINGS ###
```

The HEMCO configuration file (usually named HEMCO_Config.rc) is where you specify emissions settings. You can specify general information for HEMCO in the **SETTINGS** area at the top of the file.

- **ROOT** is the path to the HEMCO data directories (this will vary on each system, above is shown for Odyssey)
- **Logfile** specifies the name of the log file where errors, warnings, and verbose output will be sent.
- **DiagnFile** is optional. It is a file where you list which quantities you would like HEMCO to archive to netCDF files.
- **DiagnPrefix** and **DiagnFreq** specify the file template and frequency for saving HEMCO diagnostics to netCDF files.
- **Unit Tolerance** controls the amount of tolerance **(1=recommended)** if netCDF file units differ from the listed units.
- **Verbose** and **Warnings** control the amount of output that will be sent to the log file (3=most, 0=least).

HEMCO configuration file: *Base Emissions Switches*

```
# ExtNr ExtName             on/off  Species
0       Base                : on    *
    --> HEMCO_RESTART       :       true
    --> AEIC                :       true
    --> BIOFUEL             :       true
    --> BOND                :       true
    --> BRAVO               :       true
    --> CAC                 :       true
    --> XIAO                :       true
    --> C2H6                :       true
    --> EDGAR               :       true
    --> HTAP                :       false
    --> EMEP                :       true
    --> EMEP_SHIP           :       true
    --> GEIA                :       true
    --> LIANG_BROMOCARB     :       true
    --> NEI2005             :       false
    --> NEI2011             :       true
    --> RETRO               :       true
    --> SHIP                :       true
    --> NEI2011_SHIP        :       false
    --> MIX                 :       true
    --> STREETS             :       false
    --> VOLCANO             :       true
    --> RCP_3PD             :       false
    --> RCP_45              :       false
    --> RCP_60              :       false
    --> RCP_85              :       false
    --> QFED2               :       false
```

HEMCO computes emissions in two ways:

Base Emissions are emissions that can just be read from disk. Most emission inventories fall into this category.

Some emissions are implemented as **HEMCO Extensions**. In this case HEMCO has to compute these emissions using meteorological inputs (P, T, RH, U, V, etc.). More on these in the next few slides.

The list at left shows the various input options in the **Base Emissions**. This list appears at the top of the ***Base Emissions Switches*** section in the HEMCO_Config.rc file. You can toggle individual emission inventories on or off.

For example, if you wanted to turn off the **AEIC aircraft inventory,** you would change its setting under the Species column from true to false.

HEMCO configuration file: *Base Emissions Data*

```
#==============================================================================
# --- AEIC aircraft emissions ---
#
# ==> Now emit aircraft BC and OC into hydrophilic tracers BCPI and OCPI.
#==============================================================================
(((AEIC
0 AEIC_NO   $ROOT/AEIC/v2014-10/aeic_2005.geos.1x1.47L.nc  NO       2005/1-12/1/0 C xyz kg/m2/s NO   110/115 20 1
0 AEIC_CO   $ROOT/AEIC/v2014-10/aeic_2005.geos.1x1.47L.nc  CO       2005/1-12/1/0 C xyz kg/m2/s CO   110     20 1
0 AEIC_SO2  $ROOT/AEIC/v2014-10/aeic_2005.geos.1x1.47L.nc  FUELBURN 2005/1-12/1/0 C xyz kg/m2/s SO2  111     20 1
0 AEIC_SO4  -                                              -        -             - -   -       SO4  112     20 1
0 AEIC_BCPI -                                              -        -             - -   -       BCPI 113     20 1
0 AEIC_OCPI -                                              -        -             - -   -       OCPI 113     20 1
0 AEIC_ACET $ROOT/AEIC/v2014-10/aeic_2005.geos.1x1.47L.nc  HC       2005/1-12/1/0 C xyz kg/m2/s ACET 114/101 20 1
0 AEIC_ALD2 -                                              -        -             - -   -       ALD2 114/102 20 1
0 AEIC_ALK4 -                                              -        -             - -   -       ALK4 114/103 20 1
0 AEIC_C2H6 -                                              -        -             - -   -       C2H6 114/104 20 1
0 AEIC_C3H8 -                                              -        -             - -   -       C3H8 114/105 20 1
0 AEIC_CH2O -                                              -        -             - -   -       CH2O 114/106 20 1
0 AEIC_PRPE -                                              -        -             - -   -       PRPE 114/107 20 1
0 AEIC_MACR -                                              -        -             - -   -       MACR 114/108 20 1
0 AEIC_RCHO -                                              -        -             - -   -       RCHO 114/109 20 1
)))AEIC
```

Further down in the HEMCO_Config.rc file (in the ***Base Emissions Data*** section), you will find the entry for the AEIC aircraft emissions. Each of these lines indicates that you want HEMCO to read a specific species from the AEIC inventory. For each species, you list **the container name**, **the data file** to be read, the **name of the variable** in the file, the **time range**, the **units**, the **G-C species to which these emissions will be added**, any **scale factors** you wish to apply, and the **category and hierarchy** of the data.

All of the entries between the brackets **(((AEIC** and **)))AEIC** will be ignored if you turned off AEIC emissions in the ***Base Emissions Switches*** section shown on the previous slide.

HEMCO configuration file: *Base Emissions Data*

Specifying time information:

```
0 AEIC_NO   $ROOT/AEIC/v2014-10/aeic_2005.geos.1x1.47L.nc  NO  2005/1-12/1/0 C  ...
```

For each file, you can tell HEMCO when it should read data from disk.
Specify requested data times in **YEAR(S)/MONTH(S)/DAY(S)/HOUR(S)** format.

Periodicity options:
C = cycling: Use the first data year for simulation dates preceding the start of the data; or the last data year for simulation dates that occur after the end of the data.
R = Range: Only read data during the specified time range.
E = Exact: Only read data if the file timestamp exactly matches the simulation date/time.

Examples:

`2000/1/1/0 C`
Read data for date 2000/01/01 @ 00:GMT and apply it to all simulation times (cycling)

`1980-2007/1-12/1-31/0-23 R`
Read hourly for each day of the years 1980-2007. Do not read data outside of this time range.

For more information, please see:
http://wiki.geos-chem.org/The_HEMCO_User's_Guide#HEMCO_settings

HEMCO configuration file: *Extension Switches*

```
# ------------------------------------------------------------------------------
100     Custom              : off   -
101     SeaFlux             : on    DMS/ACET
102     ParaNOx             : on    NO/NO2/O3/HNO3
    --> LUT data format     :       nc
    --> LUT source dir      :       $ROOT/PARANOX/v2015-02

       . . . etc skipping some lines . . .

111     GFED                : on    NO/CO/ALK4/ACET/MEK/ALD2/PRPE/C3H8/CH2O/C2H6/SO2/NH3/BCPO/BCPI/OCPO/OCPI/POA1/NAP
    --> GFED3               :       false
    --> GFED4               :       true
    --> GFED_daily          :       false
    --> GFED_3hourly        :       false
    --> Scaling_CO          :       1.05
    --> Scaling_POG1        :       1.27
    --> Scaling POG2        :       1.27
    --> Scaling NAP         :       2.75e-4
    --> hydrophilic BC      :       0.2
    --> hydrophilic OC      :       0.5
    --> fraction POG1       :       0.49
 114 FINN                   : off   NO/CO/ALK4/ACET/MEK/ALD2/PRPE/C3H8/CH2O/C2H6/SO2/NH3/BCPI/BCPO/OCPI/OCPO/GLYC/HAC
    --> FINN_daily          :       false
    --> Scaling_CO          :       1.0
    --> hydrophilic BC      :       0.2
    --> hydrophilic OC      :       0.5
```

The ***Extension Switches Section*** (back up near the top of the HEMCO_Config.rc file) lets you select options for the various HEMCO Extensions. You specify the **extension number and name**, whether you want to **turn an extension on or off**, which **species to use with that extension**, and other **options used by the extension.**

HEMCO configuration file: *Extension data*

```
#==============================================================================
# --- GFED biomass burning emissions (Extension 111)
# NOTE: These are the base emissions in kgDM/m2/s.
#==============================================================================
111 GFED_HUMTROP    $ROOT/GFED3/v2014-10/GFED3_humtropmap.nc              humtrop           2000/1/1/0             C xy 1          * - 1 1

(((GFED3
111 GFED_WDL        $ROOT/GFED3/v2014-10/GFED3_gen.1x1.$YYYY.nc           GFED3_BB__WDL_DM  1997-2011/1-12/01/0    C xy kgDM/m2/s * - 1 1
111 GFED_AGW        $ROOT/GFED3/v2014-10/GFED3_gen.1x1.$YYYY.nc           GFED3_BB__AGW_DM  1997-2011/1-12/01/0    C xy kgDM/m2/s * - 1 1
111 GFED_DEF        $ROOT/GFED3/v2014-10/GFED3_gen.1x1.$YYYY.nc           GFED3_BB__DEF_DM  1997-2011/1-12/01/0    C xy kgDM/m2/s * - 1 1
111 GFED_FOR        $ROOT/GFED3/v2014-10/GFED3_gen.1x1.$YYYY.nc           GFED3_BB__FOR_DM  1997-2011/1-12/01/0    C xy kgDM/m2/s * - 1 1
111 GFED_PET        $ROOT/GFED3/v2014-10/GFED3_gen.1x1.$YYYY.nc           GFED3_BB__PET_DM  1997-2011/1-12/01/0    C xy kgDM/m2/s * - 1 1
111 GFED_SAV        $ROOT/GFED3/v2014-10/GFED3_gen.1x1.$YYYY.nc           GFED3_BB__SAV_DM  1997-2011/1-12/01/0    C xy kgDM/m2/s * - 1 1
)))GFED3

(((GFED4
111 GFED_WDL        $ROOT/GFED4/v2015-03/GFED4_gen.025x025.$YYYY.nc       WDL_DM            2000-2013/1-12/01/0    C xy kg/m2/s   * - 1 1
111 GFED_AGW        $ROOT/GFED4/v2015-03/GFED4_gen.025x025.$YYYY.nc       AGW_DM            2000-2013/1-12/01/0    C xy kg/m2/s   * - 1 1
111 GFED_DEF        $ROOT/GFED4/v2015-03/GFED4_gen.025x025.$YYYY.nc       DEF_DM            2000-2013/1-12/01/0    C xy kg/m2/s   * - 1 1
111 GFED_FOR        $ROOT/GFED4/v2015-03/GFED4_gen.025x025.$YYYY.nc       FOR_DM            2000-2013/1-12/01/0    C xy kg/m2/s   * - 1 1
111 GFED_PET        $ROOT/GFED4/v2015-03/GFED4_gen.025x025.$YYYY.nc       PET_DM            2000-2013/1-12/01/0    C xy kg/m2/s   * - 1 1
111 GFED_SAV        $ROOT/GFED4/v2015-03/GFED4_gen.025x025.$YYYY.nc       SAV_DM            2000-2013/1-12/01/0    C xy kg/m2/s   * - 1 1
)))GFED4

(((GFED_daily
111 GFED_FRAC_DAY   $ROOT/GFED3/v2014-10/GFED3_dailyfrac_gen.1x1.$YYYY.nc GFED3_BB__DAYFRAC 2002-2011/1-12/1-31/0  C xy 1          * - 1 1
)))GFED_daily

(((GFED_3hourly
111 GFED_FRAC_3HOUR $ROOT/GFED3/v2014-10/GFED3_3hrfrac_gen.1x1.$YYYY.nc   GFED3_BB__HRFRAC  2002-2011/1-12/01/0-23 C xy 1          * - 1 1
)))GFED_3hourly
```

Further down in the HEMCO_Config.rc file, there is an ***Extension Data Section***. In this section you list entries for data files that pertain to each of the HEMCO extensions. (You use the same syntax as for the ***Base Emissions Data*** section.).

Here the GFED extension data is shown (GFED4 is highlighted). The brackets (((GFED4 and)))GFED4 are used to turn the GFED4 emissions on or off (depending on the settings in ***Extension Switches***). Note that the number at the start of the line now is 111 and not 0, this corresponds to the GFED extension number.

HEMCO configuration file: *Scale Factors*

```
#==============================================================================
# --- temporary scale factors for comparisons
#==============================================================================
919 NO_ratio $ROOT/AnnualScalar/v2014-07/NO_ratio_2005_2002.nc NOXscalar 2005/1/1/0 C xy 1 1
918 CO_ratio $ROOT/AnnualScalar/v2014-07/CO_ratio_2005_1985.nc COscalar  2005/1/1/0 C xy 1 1

#==============================================================================
# --- day-of-week scale factors ---
# ==> data is Sun/Mon/.../Sat
#==============================================================================
20 GEIA_DOW_NOX 0.784/1.0706/1.0706/1.0706/1.0706/1.0706/0.863 - - - xy 1 1
21 GEIA_DOW_CO  0.683/1.1076/1.0706/1.0706/1.0706/1.0706/0.779 - - - xy 1 1
22 GEIA_DOW_HC  0.671/1.1102/1.1102/1.1102/1.1102/1.1102/0.768 - - - xy 1 1

#==============================================================================
# --- diurnal scale factors ---
#==============================================================================
25 EDGAR_TODNOX $ROOT/EDGARv42/v2015-02/NO/EDGAR_hourly_NOxScal.nc NOXscale 2000/1/1/HH C xy 1 1
26 GEIA_TOD_FOSSIL
0.45/0.45/0.6/0.6/0.6/0.6/1.45/1.45/1.45/1.45/1.4/1.4/1.4/1.4/1.45/1.45/1.45/1.45/0.65/0.65/0.65/0.65/0.45
/0.45 - - - xy 1 1
```

In the ***Scale Factors*** section of the HEMCO_Config.rc file, you may specify the scale factors that will be applied to the various emissions inventories listed in the ***Base Emissions Data*** and ***Extensions Data*** sections.

Scale factors can be:

- Gridded lon-lat data (read from a netCDF file), follows mostly same format as Base Emissions Data
- A list of values (i.e. global values for a sequence of years, months, or days)
- A single value

In the **units** column, "1" means the same as **"unitless"**
Scale factors also let you specify an **operation (1=multiplication, -1=division, 2=square)**

HEMCO configuration file: *Masks*

```
#==============================================================================
# Country/region masks
#==============================================================================
1000 EMEP_MASK    $ROOT/MASKS/v2014-07/EMEP_mask.geos.1x1.nc             MASK     2000/1/1/0 C xy 1 1 -30/30/45/70
1001 MEXICO_MASK  $ROOT/MASKS/v2014-07/BRAVO.MexicoMask.generic.1x1.nc  MASK     2000/1/1/0 C xy 1 1 -118/17/-95/33
1002 CANADA_MASK  $ROOT/MASKS/v2014-07/Canada_mask.geos.1x1.nc          MASK     2000/1/1/0 C xy 1 1 -141/40/-52/85
1003 SEASIA_MASK  $ROOT/MASKS/v2014-07/SE_Asia_mask.generic.1x1.nc      MASK     2000/1/1/0 C xy 1 1  60/-12/153/55
1004 NA_MASK      $ROOT/MASKS/v2014-07/NA_mask.geos.1x1.nc              MASK     2000/1/1/0 C xy 1 1 -165/10/-40/90
1005 USA_MASK     $ROOT/MASKS/v2014-07/usa.mask.nei2005.geos.1x1.nc     MASK     2000/1/1/0 C xy 1 1 -165/10/-40/90
1006 ASIA_MASK    $ROOT/MASKS/v2014-07/MIX_Asia_mask.generic.025x025.nc MASK     2000/1/1/0 C xy 1 1  46/-12/180/82
1007 NEI11_MASK   $ROOT/MASKS/v2014-07/USA_LANDMASK_NEI2011_0.1x0.1.nc  LANDMASK 2000/1/1/0 C xy 1 1 -140/20/-50/60
1008 USA_BOX      -129/25/-63/49                                         MASK     2000/1/1/0 C xy 1 1 -129/25/-63/49
```

In the ***Masks*** section, (near the end of the HEMCO_Config.rc file), you can define geographical (lon-lat) masks that are used with the regional emissions inventories. A mask is usually set to 1 where a regional inventory is used, and 0 where it is not used. Fractional values can also be specified (in v11-01 and higher).

To specify a mask for a rectangular region, you can **simply give the lon and lat of at the lower left and upper right corners**. For irregularly-shaped regions, it is better to create a netCDF file for the mask.

For each mask, you specify the following:

- **A number and container name for the mask**
- **The file where the mask resides (optional)**
- **The name of the variable in the file**
- **The time range**
- **The units ("1" = "unitless")**
- **The operation you want to do: 1=multiplication (most common), -1=division, 3=invert mask**
- **The longitude and latitude range of the mask (only used for MPI environments)**

HEMCO configuration file: *Non-Emissions Data*

```
# --- Time zones (offset to UTC) ---
* TIMEZONES $ROOT/TIMEZONES/v2015-02/timezones_1x1.nc UTC_OFFSET 2000/1/1/0 C xy count * - 1 1

# --- UV albedo, for photolysis (cf Hermann & Celarier, 1997) ---
(((+UValbedo+
* UV_ALBEDO $ROOT/UVALBEDO/v2015-03/uvalbedo.geos.2x25.nc UVALBD 1985/1-12/1/0 C xy 1 * - 1 1
)))+UValbedo+

# --- TOMS/SBUV overhead ozone colums, for photolysis ---
(((+TOMS_SBUV_O3+
* TOMS_O3_COL   $ROOT/TOMS_SBUV/v2015-03/TOMS_O3col_$YYYY.geos.1x1.nc TOMS   1971-2010/1-12/1/0 C xy dobsons     * - 1 1
* DTOMS1_O3_COL -                                                     DTOMS1 -                   - -  dobsons/day * - 1 1
* DTOMS2_O3_COL -                                                     DTOMS2 -                   - -  dobsons/day * - 1 1
)))+TOMS_SBUV_O3+

# --- Linear stratospheric chemistry fields ---
# These fields will only be read if the +LinStratChem+ toggle is activated.

(((+LinStratChem+

# --- Stratospheric Bry data from the CCM model  ---
* GEOSCCM_Br_DAY      $ROOT/STRAT/v2015-01/Bry/GEOSCCM_Bry.2007$MM.day.nc   BR     2007/$MM/1/0 C xyz pptv * - 60 1
* GEOSCCM_Br_NIGHT    $ROOT/STRAT/v2015-01/Bry/GEOSCCM_Bry.2007$MM.night.nc BR     2007/$MM/1/0 C xyz pptv * - 60 1
... etc...


#--- GMI chemistry: prod/loss rates (for strato-/mesosphere) ---
* GMI_LOSS_A3O2       $ROOT/GMI/v2015-02/gmi.clim.A3O2.geos5.2x25.nc        loss    2000/$MM/1/0 C xyz s-1   A3O2     - 1  1
* GMI_PROD_A3O2       $ROOT/GMI/v2015-02/gmi.clim.A3O2.geos5.2x25.nc        prod    2000/$MM/1/0 C xyz v/v/s A3O2     - 1  1
... etc ...
)))+LinStratChem+
```

The ***Non-Emissions Data*** section is similar to ***Base Emissions Data*** section, with a couple of differences:

- The switches +UValbedo+ and +TOMS_SBUV_O3+ will be automatically set if FAST-JX is turned on.
- The switch +LinStratChem+ will be automatically set if stratospheric chemistry is turned on.

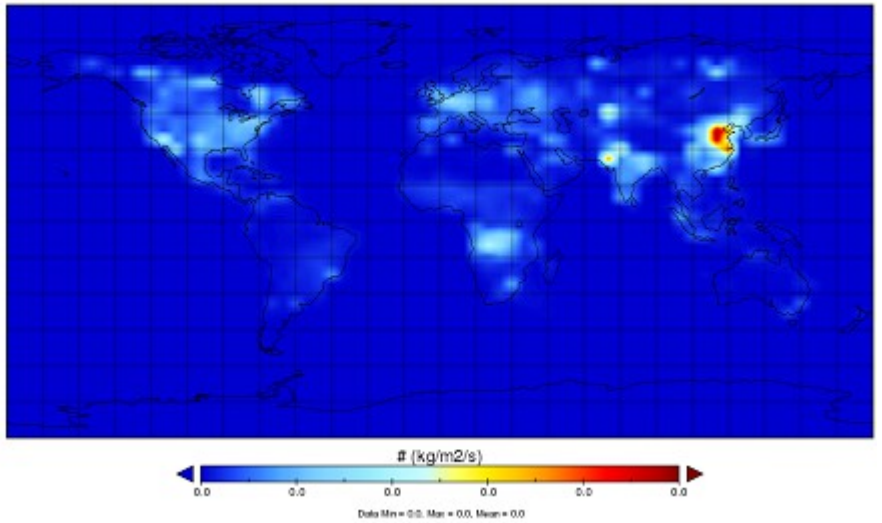
This prevents errors caused by inconsistent input between `input.geos` and `HEMCO_Config.rc`.

You must manually retrieve non-emissions data into your GEOS-Chem routine with a call to `HCO_GetPtr.`

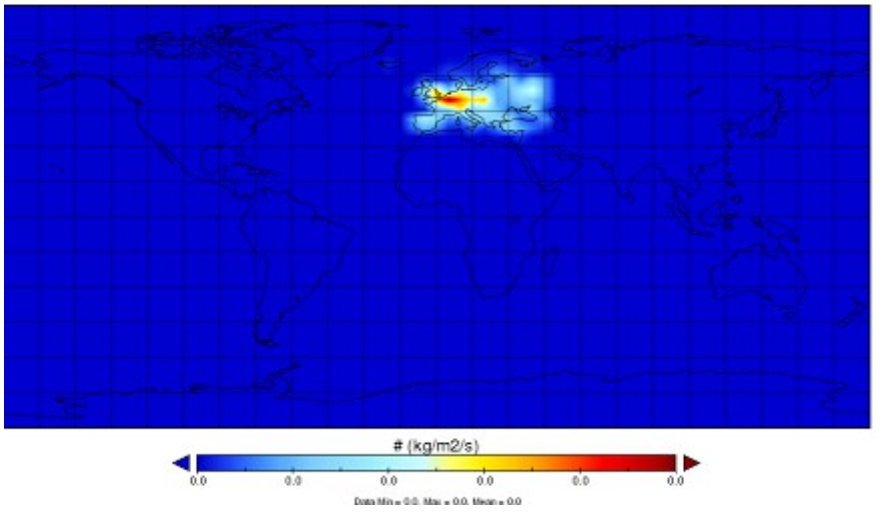
```
# Sample HEMCO_diagnostics.rc file
#
# This is the file specified in the “DiagnFile” slot of HEMCO_Config.rc.
# It specifies which HEMCO quantities you would like to archive to netCDF files.
#
# The netCDF file names begin with “DiagnPrefix” (from HEMCO_Config.rc),
# and will be archived at frequency “DiagnFreq” (also from HEMCO_Config.rc).
#
# NOTE: Category 1/2 means to put all of the emissions into category #1
# (which is anthropogenic) and zero category #2 (which is biomass).
# This is needed for those inventories that lump anthro + biomass together.
#
# Name        Spec ExtNr Cat Hier Dim Unit

TOTAL_NO      NO    -1    -1  -1   2   kg/m2/s   # NO from all sources

EMEP_NO       NO     0   1/2  10   2   kg/m2/s   # NO just from EMEP

GFED_CO       CO   111     5  -1   2   kg/m2/s   # CO from GFED biomass burning

# If you want to just diagnose regional emissions, then you need to set the
# diagnostics extension number, category and hierarchy accordingly:
#
NEI2011_CO    CO     0     1  50   2   kg/m2/s   # CO just from NEI2011
```

Output from the HEMCO_Diagnostic_201307020000.nc file

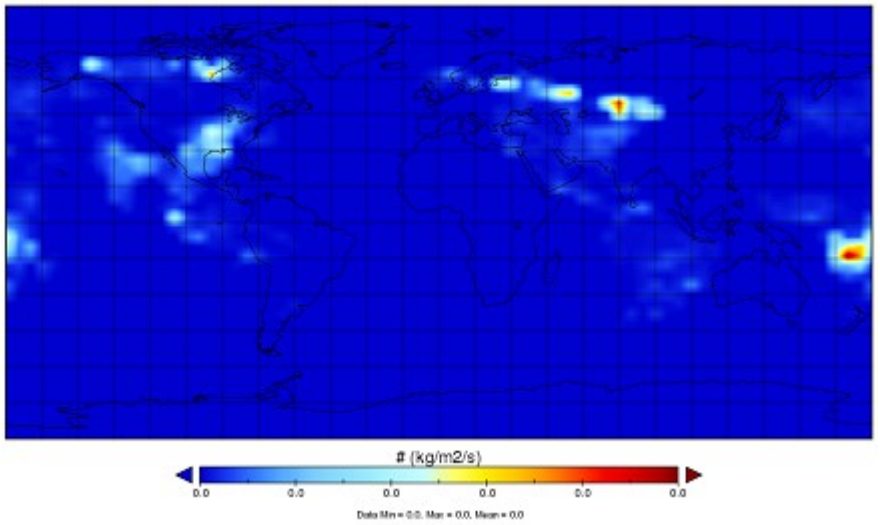
TOTAL_NO



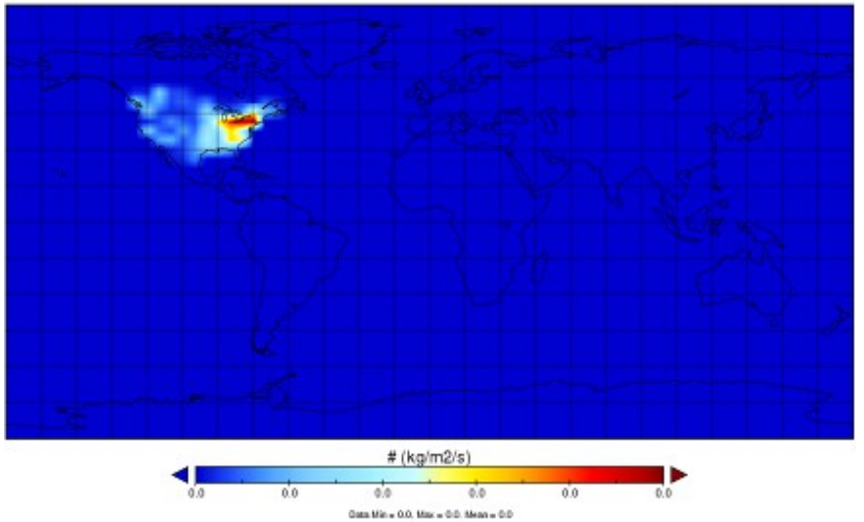
EMEP_NO



GFED_CO



NEI2011_CO



HEMCO standalone mode

- When you compile GEOS-Chem, it creates the HEMCO standalone executable in folder:
  - `bin/hemco_standalone.x`
- You can use the HEMCO standalone to test new emissions inventories without having to run them in GEOS-Chem
- For more information:
  - **wiki.geos-chem.org/The_HEMCO_User's_Guide#Stand-alone_interface**

FlexChem:
A clean implementation of the KPP chemical solver

Introduced in v11-01

FlexChem overview

- FlexChem completely removed SMVGEAR and all related code (i.e. no more `CSPEC`, `JLOP`, etc).
- Based on the KPP (Kinetic Pre-Processor) code by Adrian Sandu et al
- In the future we may be able to upgrade to Kppa (KPP-Accelerated) by ParaTools (John Linford)
- For more information:
  - **wiki.geos-chem.org/FlexChem**

FlexChem chemistry mechanisms

- Choose from 5 "out-of-the-box" mechanisms:
  1) Tropchem
     - NOx + Ox + HC + Aerosols + Bromine, troposphere only
  2) UCX
     - Adds stratospheric chemistry, chlorine species
  3) SOA
  4) SOA + semi-volatile POA (aka SOA-SVPOA)
  5) Standard (= Tropchem + UCX + SOA)
     - **This is what we use for the 1-month & 1-year benchmarks**

Updating a FlexChem mechanism

- You must modify 3 input files for the KPP solver:
  - `gckpp.kpp`
    - Defines integrator, P/L families, and inlined rate functions
  - `globchem.spc`
    - Defines the list of variable and fixed species
  - `globchem.eqn`
    - Defines each of the reactions in the mechanism
    - Specifies the rate function corresponding to each reaction
      - Gas-phase rates are defined in **`gckpp.kpp`**
      - Hetchem rates are defined instead in **`gckpp_HetRates.F90`**

```
#INTEGRATOR rosenbrock
#LANGUAGE Fortran90
#DRIVER none
#HESSIAN off
#MEX off
#STOICMAT off

#INCLUDE globchem.spc
#INCLUDE globchem.eqn

#FAMILIES
POx : O3 + NO2 + 2NO3 + PAN + PMN + PPN + HNO4 + 3N2O5 + HNO3 + BrO + HOBr
+ BrNO2 + 2BrNO3 + MPN + ETHLN + ISN1 + ISOPNB + ISOPND + MACRN + MVKN +
PROPNN + R4N2 + INPN + ISNP + INO2 + ISNOOA + ISNOOB + ISNOHOO + MAN2 +
PRN1 + PRPN + R4N1 + PMNN + MACRNO2 + ClO + HOCl + ClNO2 + 2ClNO3 + 2Cl2O2
+ 2OClO + O + O1D;
LOx : O3 + NO2 + 2NO3 + PAN + PMN + PPN + HNO4 + 3N2O5 + HNO3 + BrO + HOBr
+ BrNO2 + 2BrNO3 + MPN + ETHLN + ISN1 + ISOPNB + ISOPND + MACRN + MVKN +
PROPNN + R4N2 + INPN + ISNP + INO2 + ISNOOA + ISNOOB + ISNOHOO + MAN2 +
PRN1 + PRPN + R4N1 + PMNN + MACRNO2 + ClO + HOCl + ClNO2 + 2ClNO3 + 2Cl2O2
+ 2OClO + O + O1D;
PCO : CO;
LCO : CO;
PSO4 : SO4;

#INLINE F90_RATES
  REAL(kind=dp) FUNCTION OH_O1D (J, H2O, TEMP, NUMDEN)
  ... etc ...
```

gckpp.kpp: Defines the following quantities:

- Integrator (e.g. Rosenbrock, Runge-Kutta, etc.)
- Language (this is always “Fortran90”)
- Prod/Loss families for GC diagnostics.
- Inlined Fortran functions to compute rate constants for reactions

```
#include atoms

#DEFVAR

      A3O2            =         IGNORE;
      ACET            =         IGNORE;
      ALD2            =         IGNORE;
      ALK4            =         IGNORE;
      ATO2            =         IGNORE;
      B3O2            =         IGNORE;
      BENZ            =         IGNORE;
      C2H6            =         IGNORE;
      C3H8            =         IGNORE;
      CH2O            =         IGNORE;
      CH4             =         IGNORE;
       ... etc ...

#DEFFIX

      ACTA            =         IGNORE;
      EMISSION        =         IGNORE;
      EOH             =         IGNORE;
      H2              =         IGNORE;
      HCOOH           =         IGNORE;
      MNO3            =         IGNORE;
      MOH             =         IGNORE;
      N2              =         IGNORE;
      NH2             =         IGNORE;
      NH3             =         IGNORE;
      O2              =         IGNORE;
      RCOOH           =         IGNORE;
```

globchem.spc

Variable (aka active) species go under the #DEFVAR heading.

Fixed (aka inactive) species go under the #DEFFIX heading.

globchem.eqn
Reactions are listed along with the Fortran functions that compute their rates.
Gas-phase rate functions are in **gckpp.kpp** and hetchem functions in **gckpp_HetRates.F90**.
Comments are contained between {} characters. Equation numbers removed from v11-02.

```
#Equations
{1} O3 + NO = NO2 + O2 :                          GCARR(3.00E-12, 0.0E+00, -1500.0);
{2} O3 + OH = HO2 + O2 :                          GCARR(1.70E-12, 0.0E+00, -940.0);
{3} O3 + HO2 = OH + O2 + O2 :                     GCARR(1.00E-14, 0.0E+00, -490.0);
{4} O3 + NO2 = O2 + NO3 :                         GCARR(1.20E-13, 0.0E+00, -2450.0);
{5} O3 + MO2 = CH2O + HO2 + O2 :                  GCARR(2.90E-16, 0.0E+00, -1000.0);
{6} OH + OH = H2O + O :                           GCARR(1.80E-12, 0.0E+00, 0.0);
{7} OH + OH {+M} = H2O2 :                         GCJPLPR(6.90E-31, 1.0E+00, 0.0, ...
{8} OH + HO2 = H2O + O2 :                         GCARR(4.80E-11, 0.0E+00, 250.0);
{9} OH + H2O2 = H2O + HO2 :                       GCARR(1.80E-12, 0.0E+00, 0.0);
{10} HO2 + NO = OH + NO2 :                        GCARR(3.30E-12, 0.0E+00, 270.0);
{11} HO2 + HO2 = H2O2 + O2 :                      GC_HO2NO3(3.00E-13, 0.0E+00, ...
{12} CO + OH = HO2 + CO2 :                        GC_OHCO(1.50E-13, 0.0E+00, 0.0);
{13} OH + CH4 = MO2 + H2O :                       GCARR(2.45E-12, 0.0E+00, -1775.0);
{14} MO2 + NO = CH2O + HO2 + NO2 :                GCARR(2.80E-12, 0.0E+00, 300.0);
{15} MO2 + HO2 = MP + O2 :                        GCARR(4.10E-13, 0.0E+00, 750.0);
{16} MO2 + MO2 = MOH + CH2O + O2 :                GC_TBRANCH(9.50E-14, 0.0E+00, ...
{17} MO2 + MO2 = 2.000CH2O + 2.000HO2 :           GC_TBRANCH(9.50E-14, 0.0E+00, ...
{18} MP + OH = MO2 + H2O :                        GCARR(2.66E-12, 0.0E+00, 200.0);
{19} MP + OH = CH2O + OH + H2O :                  GCARR(1.14E-12, 0.0E+00, 200.0);
{20} ATOOH + OH = ATO2 + H2O :                    GCARR(2.66E-12, 0.0E+00, 200.0);
{21} ATOOH + OH = MGLY + OH + H2O :               GCARR(1.14E-12, 0.0E+00, 200.0);
{22} CH2O + OH = CO + HO2 + H2O :                 GCARR(5.50E-12, 0.0E+00, 125.0);
{23} NO2 + OH {+M} = HNO3 {+M} :                  GCJPLPR(1.80E-30, 3.0E+00, 0.0, ...
... etc ...
```

Updating FlexChem mechanisms

1) Download KPP source code from Bitbucket.org
2) Compile KPP to an executable
3) Update **`gckpp.kpp`**
4) Update **`globchem.spc`** (if adding a new species)
5) Update **`globchem.eqn`** (change rates, coeffs, etc)
6) Run the KPP executable to create Fortran code
7) Copy Fortran code to `KPP/Custom` folder of GC
8) Compile GC with `CHEM=Custom` flag

Updating FlexChem mechanisms

- Thankfully, Melissa has added very detailed information to the FlexChem wiki page
- Follow these instructions:
  - **wiki.geos-chem.org/FlexChem#Adding_chemical_mechanisms_in_FlexChem**

GEOS-Chem Species Database: A one-stop shop for species physical properties

Introduced in v11-01

GEOS-Chem Species Database

- The species database is a vector of derived-type objects stored within `State_Chm`
- Each element in the vector is an object of type `SPECIES` (defined in `species_mod.F90`)
- Each `SPECIES` object contains information about a single GEOS-Chem species:
  - Indices and inquiry variables (i.e. is it advected?)
  - Molecular weights and Henry's law constants
  - Wetdep, drydep, aerosol, & other relevant parameters

The **species database** is implemented as a sub-field of the **`State_Chm`** object.

`State_Chm` also carries several **counters** and **mapping arrays** that are used in conjunction with the **species database**.

```
!=========================================================================
! Derived type for Chemistry State
!=========================================================================
TYPE, PUBLIC :: ChmState

   ! Number of species
   INTEGER                   :: nSpecies            ! # of species
   INTEGER                   :: nAdvect             ! # of advected species
   INTEGER                   :: nDrydep             ! # of drydep species
   INTEGER                   :: nKppSpc             ! # of KPP chem species
   INTEGER                   :: nWetDep             ! # of wetdep species

   ! Mapping vectors to subset types of species
   INTEGER,          POINTER :: Map_Advect (:     ) ! Advected species ID's
   INTEGER,          POINTER :: Map_DryDep (:     ) ! Drydep species ID's
   INTEGER,          POINTER :: Map_KppSpc (:     ) ! KPP chem species ID's
   INTEGER,          POINTER :: Map_WetDep (:     ) ! Wetdep species IDs'

   ! Physical properties about tracers & species
   TYPE(SpcPtr),     POINTER :: SpcData(:)          ! Species database

   ... etc ...

END TYPE ChmState
```

Obtaining information from the species database [edit]

To get the physical properties for a given species from the species database, you can use code such as:

```
      DO N = 1, State_Chm%nSpecies

         ! Get Henry's law parameters
         K0  = State_Chm%SpcData(N)%Info%Henry_K0
         CR  = State_Chm%SpcData(N)%Info%Henry_CR
         pKA = State_Chm%SpcData(N)%Info%Henry_pKa

      ENDDO
```

To simplify the coding, you can create an object of type `Species` to point to each entry in the species database. For example this code

```
      USE Species_Mod, ONLY : Species

      TYPE(Species), POINTER :: ThisSpc

      DO N = 1, State_Chm%nSpecies

         ThisSpc => State_Chm%SpcData(N)%Info

         ! Get Henry's law parameters
         K0  =  ThisSpc%Henry_K0
         CR  =  ThisSpc%Henry_CR
         pKA =  ThisSpc%Henry_pKa

         ThisSpc => NULL()

      ENDDO
```

is much less wordy and easier to read.

Adding species to the GC Species DB

To add a new species, add the relevant information into GEOS-Chem module `Headers/species_database_mod.F90`

Example 2: Add a gas-phase species that is advected, dry-deposited, and wet-deposited [edit]

Hydrogen peroxide is an example of a GEOS-Chem species that is advected, dry-deposited, and wet-deposited. It is defined with the following settings:

```
CASE( 'H2O2' )
   CALL Spc_Create( am_I_Root     = am_I_Root,                &
                    ThisSpc       = SpcData(N)%Info,          &
                    ModelID       = N,                        &
                    Name          = NameAllCaps,              &
                    FullName      = 'Hydrogen peroxide',      &
                    MW_g          = 34.0_fp,                  &
                    Is_Advected   = T,                        &
                    Is_Gas        = T,                        &
                    Is_Drydep     = T,                        &
                    Is_Wetdep     = T,                        &
                    DD_F0         = 1.0_fp,                   &
                    DD_Hstar_old  = 1e+5_fp,                  &
                    Henry_K0      = 8.30e+4_f8,               &
                    Henry_CR      = 7400.0_f8,                &
                    WD_RetFactor  = 5e-2_fp,                  &
                    WD_LiqAndGas  = T,                        &
                    WD_ConvFacI2G = 4.36564e-1_fp,            &
                    RC            = RC )
```

For more information about the properties that can be stored in the Species Database, see:
http://wiki.geos-chem.org/GEOS-Chem_species_database
http://wiki.geos-chem.org/Physical_properties_of_GEOS-Chem_species

Adding species to the GC Species DB

There are several options you can specify when you add a new species:
Gas or aerosol? Is it advected? Is it dry-deposited? Is it wet-deposited?
Is it carried in moles of carbon?

Example 4: Add a hydrocarbon species carried as equivalent carbon atoms [edit]

Several GEOS-Chem hydrocarbon species (e.g. acetone, isoprene, etc.) are emitted and transported as equivalent carbon atoms instead of a single molecule.

```
CASE( 'ACET' )
   CALL Spc_Create( am_I_Root     = am_I_Root,                &
                    ThisSpc       = SpcData(N)%Info,          &
                    ModelID       = N,                        &
                    Name          = NameAllCaps,              &
                    FullName      = 'Acetone',                &
                    MW_g          = 58.08_fp,                 &
                    EmMW_g        = 12.00_fp,                 &
                    MolecRatio    = 3.0_fp,                   &
                    Is_Advected   = T,                        &
                    Is_Gas        = T,                        &
                    Is_Drydep     = T,                        &
                    Is_Wetdep     = F,                        &
                    DD_F0         = 1.0_fp,                   &
                    DD_Hstar_Old  = 1e5_fp,                   &
                    Henry_K0      = 2.7e+1_f8,                &
                    Henry_CR      = 5300.0_f8,                &
                    RC            = RC )
```

```
SUBROUTINE MySub( ..., State_Chm, ... )

     !%%% Example of the new indexing scheme %%%

     ! Uses
     USE State_Chm_Mod, ONLY : Ind_

     ! Local variables
     INTEGER  :: id_O3, id_Br2, id_CO

     ! Find species indices with the Ind_() function
     id_O3  = Ind_( 'O3'  )
     id_Br2 = Ind_( 'Br2' )
     id_CO  = Ind_( 'CO'  )

     ! Print concentrations
     ! NOTE: Should test to see if indices are valid, omitted here!!
     print*, 'O3  at (23,34,1) : ', State_Chm%Species(23,34,1,id_O3 )
     print*, 'Br2 at (23,34,1) : ', State_Chm%Species(23,34,1,id_Br2)
     print*, 'CO  at (23,34,1) : ', State_Chm%Species(23,34,1,id_CO )

     ! Print the molecular weight of O3
     ! obtained from the Species Database
     print*, 'Mol wt of O3 [g]: ', State_Chm%SpcData(id_O3)%Info%MW_g

  END SUBROUTINE MySub
```

New method: function `Ind_()`

This function returns the species index that is consistent with the ordering in the GEOS-Chem species database.

This is also consistent with the chemical species ordering for FlexChem.

New method: function `Ind_()`, continued

By default, **`Ind_()`** returns the overall species index.

You can supply optional arguments to **`Ind_()`** to return other species indices, as shown below:

```
! Position of HNO3 in the list of advected species
AdvectId = Ind( 'HNO3', 'A' )

! Position of HNO3 in the list of dry deposited species
DryDepId = Ind( 'HNO3', 'D' )

! Position of HNO3 in the list of wet deposited species
WetDepId = Ind( 'HNO3', 'W' )

! Position of HNO3 in the list of fixed KPP species
KppFixId = Ind( 'HNO3', 'F' )

! Position of HNO3 in the list of variable KPP species
KppVarId = Ind( 'HNO3', 'V' )
```

For more information

- All on the GEOS-Chem wiki:
  - HEMCO
  - Implementation of HEMCO in GEOS-Chem
  - The HEMCO User's Guide
  - FlexChem
  - GEOS-Chem species database
  - Physical properties of GEOS-Chem species
- Email us:
  - `geos-chem-support@as.harvard.edu`

For more information

- GC Adjoint model clinic TODAY @ 5:15 PM
- GCHP model clinic Tuesday @ 12:45 PM
- GC/ESM model clinic Wednesday @ 12:45 PM
- GCST office hours
  - 1:30 – 3:00 PM Mon thru Weds, in the lobby
    (All the way down the hall, near the Oxford St. exit)