GEOS-5 AGCM Gridded Components

GEOS-5 developers

March 14, 2014

Contents

1 Module ARIESg3_GridCompMod — ARIESr/GEOS3 Dynamical Core Grid Component

USES:

```
use ESMF                 ! ESMF base class
use MAPL_Mod             ! GEOS base class
use dynamics_vars, only : T_TRACERS, T_FVDYCORE_VARS, &
                          T_FVDYCORE_GRID, T_FVDYCORE_STATE
```

PUBLIC MEMBER FUNCTIONS:

```
public  SetServices      ! Register component methods
```

DESCRIPTION:
This module implements the FVCAM Dynamical Core as an ESMF gridded component.

Overview This module contains an ESMF wrapper for the Finite-Volume Dynamical Core used in the Community Atmospheric Model (FVCAM). This component will hereafter be referred to as the "FVdycore" ESMF gridded component. FVdycore consists of four subcomponents,

- `cd_core`: The C/D-grid dycore component
- `te_map`: Vertical remapping algorithm
- `trac2d`: Tracer advection
- `benergy`: Energy balance

Subsequently the ESMF component design for FV dycore will be described.

Internal State FVdycore maintains an internal state consisting of the following fields: control variables

- `U`: U winds on a D-grid (m/s)
- `V`: V winds on a D-grid (m/s)
- `PT`: Scaled Virtual Potential Temperature (T_v/PKZ)
- `PE`: Edge pressures
- `Q`: Tracers
- `PKZ`: Consistent mean for p^κ

as well as a GRID (to be mentioned later) and same additional run-specific variables (dt, iord, jord, nsplit – to be mentioned later)
Note: `PT` is not updated if the flag `CONVT` is true.
The internal state is updated each time FVdycore is called.

Import State The import state consists of the tendencies of the control variables plus the surface geopotential heights:

- DUDT: U wind tendency on a A-grid (m/s)
- DVDT: V wind tendency on a A-grid (m/s)
- DTDT: Delta-pressure-weighted temperature tendency
- DPEDT: Edge pressure tendency
- PHIS: Surface Geopotential Heights

These are by definition on an A-grid and have an XY domain decomposition.

Export State The export state can provide the following variables:

- U: U winds on a A-grid (m/s)
- V: V winds on a A-grid (m/s)
- U_CGRID: U winds on a C-grid (m/s)
- V_CGRID: V winds on a C-grid (m/s)
- U_DGRID: U winds on a D-grid (m/s)
- V_DGRID: V winds on a D-grid (m/s)
- T: Temperature (K)
- Q: Tracers
- TH: Potential Temperature (K)
- ZL: Mid-Layer Heights (m)
- ZLE: Edge Heights (m)
- PLE: Edge pressures (Pa)
- PLK: P^{κ} at Mid-Layers
- OMEGA: Vertical pressure velocity (pa/s)
- PTFX: Mass-Weighted PT flux on C-Grid (K Pa m^2/s)
- PTFY: Mass-Weighted PT flux on C-Grid (K Pa m^2/s)
- MFX_UR: Mass-Weighted U-Wind on C-Grid (Pa m^2/s)
- MFY_UR: Mass-Weighted V-wind on C-Grid (Pa m^2/s)

- `MFX`: Remapped Mass-Weighted U-Wind on C-Grid (Pa m^2/s)
- `MFY`: Remapped Mass-Weighted V-wind on C-Grid (Pa m^2/s)
- `MFZ`: Remapped Vertical mass flux (kg/(m^2*s))
- `MFX_A`: Remapped Mass-Weighted U-Wind on A-Grid (Pa m^2/s)
- `MFY_A`: Remapped Mass-Weighted V-wind on A-Grid (Pa m^2/s)
- `PV`: Ertel's Potential Vorticity (m^2 / kg*s)
- `DUDT`: U-wind Tendency (m/s/s)
- `DVDT`: V-wind Tendency (m/s/s)
- `DTDT`: Mass-Weighted Temperature Tendency (Pa K/s)
- `AREA`: Cell areas on the A-Grid (m^2, polar caps at $J = 1$, $J = JM$)

All variables are on an A-grid with points at the poles, and have an XY decomposition.

Grids and Decompositions The current version supports only a 1D latitude-based decomposition of the domain (with OMP task-parallelism in the vertical, resulting in reasonable scalability on large PE configurations). In the near future it will support a 2D domain decomposition, in which import and export state are decomposed in longitude and latitude, while the internal state (for the most part) is decomposed in latitude and level. When needed, the data is redistributed ("transposed") internally.
There are two fundamental ESMF grids in use;

- GRIDXY: longitude-latitude ESMF grid (public)
- GRIDYZ: A latitude-level cross-sectional decomposition (private to this module)

PILGRIM will be used for communication until ESMF has sufficient functionality and performance to take over the task. The use of pilgrim requires a call to `INIT_SPMD` to set SPMD parameters, decompositions, etc.
Currently, only a 1D decomposition in latitude is employed. Thus GRIDXY and GRIDYZ actually represent the same decomposition and no transposes are employed.

Required Files The following files are needed for a standard restart run:

- Layout file
  - `nprxy_x, nprxy_y, npryz_y, npryz_z`: process dimensions in XY and YZ.
  - `imxy, jmxy, jmyz, kmyz`: distributions for XY and YZ
  - `iord, jord`: the order of the lon. and lat. algorithms
  - `dtime`: The large (advection) time step

  - `nsplit`: the ratio between the large and small time step (possibly zero for automatic determination),

- Restart file
  - date in standard format yy, mm, dd, hh, mm, ss
  - dimensions im, jm, km, nq
  - control variables `U, V, PT, PE, Q`

- Topography file

Future Additions

- Conservation of energy (CONSV = = .TRUE.)
- 2D decomposition (requires transposes in the coupler)
- Use r8 instead of r4 (currently supported in StopGap)

1.1 SetServices — Set services for FVCAM Dynamical Core

INTERFACE:

```
Subroutine SetServices ( gc, rc )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: gc     ! gridded component
integer, intent(out), optional     :: rc     ! return code
```

DESCRIPTION:
Set services (register) for the FVCAM Dynamical Core Grid Component. *STATES:*

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
DUDT	IM	$m\ s^{-2}$	xyz	Center	eastward wind tendency
DVDT	IM	$m\ s^{-2}$	xyz	Center	northward wind tendency
DTDT	IM	$Pa\ K\ s^{-1}$	xyz	Center	delta-p weighted temperature tendency
DQVANA	IM	$kg\ kg^{-1}$	xyz	Center	specific humidity increment from analysis

Short Name	Type	Units	Dims	Vert Loc	Long name
DOXANA	IM	kg kg^{-1}	xyz	Center	ozone increment from analysis
DPEDT	IM	Pa s^{-1}	xyz	Edge	edge pressure tendency
PHIS	IM	m^2 sec^{-2}	xy	None	surface geopotential height
TRADV	IM	unknown			advected quantities
KE	EX	J m^{-2}	xy	None	vertically integrated kinetic energy
TAVE	EX	K	xy	None	vertically averaged dry temperature
UAVE	EX	m sec^{-1}	xy	None	vertically averaged zonal wind
KEPHY	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to physics
PEPHY	EX	W m^{-2}	xy	None	total potential energy tendency due to physics
TEPHY	EX	W m^{-2}	xy	None	mountain work tendency due to physics
KEANA	EX	W m^{-2}	xy	None	total kinetic energy tendency due to analysis
PEANA	EX	W m^{-2}	xy	None	total potential energy tendency due to analysis
TEANA	EX	W m^{-2}	xy	None	mountain work tendency due to analysis
KEHOT	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to HOT
KEDP	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to pressure change
KEADV	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to dynamics advection
KEPG	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to pressure gradient
KEDYN	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to dynamics
PEDYN	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency due to dynamics
TEDYN	EX	W m^{-2}	xy	None	mountain work tendency due to dynamics
KECDCOR	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to cdcore

Short Name	Type	Units	Dims	Vert Loc	Long name
PECDCOR	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency due to cdcore
TECDCOR	EX	W m^{-2}	xy	None	mountain work tendency due to cdcore
QFIXER	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency due to CONSV
KEREMAP	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to remap
PEREMAP	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency due to remap
TEREMAP	EX	W m^{-2}	xy	None	mountain work tendency due to remap
KEGEN	EX	W m^{-2}	xy	None	vertically integrated generation of kinetic energy
DKERESIN	EX	W m^{-2}	xy	None	vertically integrated kinetic energy residual from inertial terms
DKERESPG	EX	W m^{-2}	xy	None	vertically integrated kinetic energy residual from PG terms
DMDTANA	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated mass tendency due to analysis
DOXDTANAINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated ozone tendency due to analysis
DQVDTANAINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated water vapor tendency due to analysis
DQLDTANAINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated liquid water tendency due to analysis
DQIDTANAINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated ice water tendency due to analysis
DMDTDYN	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated mass tendency due to dynamics
DOXDTDYNINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated ozone tendency due to dynamics
DTHVDTDYNINT	EX	K kg m^{-2} s^{-1}	xy	None	vertically integrated THV tendency due to dynamics
DTHVDTREMAP	EX	K kg m^{-2} s^{-1}	xy	None	vertically integrated THV tendency due to vertical remapping

Short Name	Type	Units	Dims	Vert Loc	Long name
DTHVDTCONSV	EX	K kg m^{-2} s^{-1}	xy	None	vertically integrated THV tendency due to TE conservation
DTHVDTPHYINT	EX	K kg m^{-2} s^{-1}	xy	None	vertically integrated THV tendency due to physics
DTHVDTANAINT	EX	K kg m^{-2} s^{-1}	xy	None	vertically integrated THV tendency due to analysis
DQVDTDYNINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated water vapor tendency due to dynamics
DQLDTDYNINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated liquid water tendency due to dynamics
DQIDTDYNINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated ice water tendency due to dynamics
CONVKE	EX	W m^{-2}	xy	Center	vertically integrated kinetic energy convergence
CONVTHV	EX	W m^{-2}	xy	Center	vertically integrated thetav convergence
CONVCPT	EX	W m^{-2}	xy	Center	vertically integrated enthalpy convergence
CONVPHI	EX	W m^{-2}	xy	Center	vertically integrated geopotential convergence
U	EX	m s^{-1}	xyz	Center	eastward wind
V	EX	m s^{-1}	xyz	Center	northward wind
T	EX	K	xyz	Center	air temperature
PL	EX	Pa	xyz	Center	mid level pressure
ZLE	EX	m	xyz	Edge	edge heights
ZL	EX	m	xyz	Center	mid layer heights
S	EX	m	xyz	Center	mid layer dry static energy
PLE	EX	Pa	xyz	Edge	edge pressure
TH	EX	K	xyz	Center	potential temperature
PLK	EX	Pa^{κ}	xyz	Center	mid-layer p^{κ}
OMEGA	EX	Pa sec^{-1}	xyz	Center	vertical pressure velocity
PTFX	EX	K Pa m^2 s^{-1}	xyz	Center	pressure weighted eastward potential temperature flux unremapped
PTFY	EX	K Pa m^2 s^{-1}	xyz	Center	pressure weighted northward potential temperature flux unremapped
MFX_UR	EX	Pa m^2 s^{-1}	xyz	Center	pressure weighted eastward wind unremapped
MFY_UR	EX	Pa m^2 s^{-1}	xyz	Center	pressure weighted northward wind unremapped

Short Name	Type	Units	Dims	Vert Loc	Long name
MFX	EX	Pa m^2 s^{-1}	xyz	Center	pressure weighted eastward wind
MFY	EX	Pa m^2 s^{-1}	xyz	Center	pressure weighted northward wind
MFZ	EX	kg m^{-2} s^{-1}	xyz	Edge	vertical mass flux
MFX_A	EX	Pa m^2 s^{-1}	xyz	Center	zonal mass flux
MFY_A	EX	Pa m^2 s^{-1}	xyz	Center	meridional mass flux
PV	EX	m^2 kg^{-1} sec^{-1}	xyz	Center	ertels isentropic potential vorticity
EPV	EX	K m^2 kg^{-1} sec^{-1}	xyz	Center	ertels potential vorticity
Q	EX	1	xyz	Center	specific humidity
DUDTANA	EX	m/sec/sec	xyz	Center	tendency of eastward wind due to analysis
DVDTANA	EX	m/sec/sec	xyz	Center	tendency of northward wind due to analysis
DTDTANA	EX	K sec^{-1}	xyz	Center	tendency of air temperature due to analysis
DDELPDTANA	EX	K sec^{-1}	xyz	Center	tendency of pressure thickness due to analysis
DUDTDYN	EX	m/sec/sec	xyz	Center	tendency of eastward wind due to dynamics
DVDTDYN	EX	m/sec/sec	xyz	Center	tendency of northward wind due to dynamics
DTDTDYN	EX	K sec^{-1}	xyz	Center	tendency of air temperature due to dynamics
DQVDTDYN	EX	kg/kg/sec	xyz	Center	tendency of specific humidity due to dynamics
DQIDTDYN	EX	kg/kg/sec	xyz	Center	tendency of ice water due to dynamics
DQLDTDYN	EX	kg/kg/sec	xyz	Center	tendency of liquid water due to dynamics
DOXDTDYN	EX	kg/kg/sec	xyz	Center	tendency of ozone due to dynamics
PREF	EX	Pa	z	Edge	reference air pressure
PS	EX	Pa	xy	None	surface pressure
TA	EX	K	xy	None	surface air temperature
QA	EX	kg kg^{-1}	xy	None	surface specific humidity
US	EX	m s^{-1}	xy	None	surface eastward wind
VS	EX	m s^{-1}	xy	None	surface northward wind
SPEED	EX	m s^{-1}	xy	None	surface wind speed
DZ	EX	m	xy	None	surface layer height
SLP	EX	Pa	xy	None	sea level pressure

Short Name	Type	Units	Dims	Vert Loc	Long name
H1000	EX	m	xy	None	height at 1000 mb
TROPP_EPV	EX	Pa	xy	None	tropopause pressure based on EPV estimate
TROPP_THERMAL	EX	Pa	xy	None	tropopause pressure based on thermal estimate
TROPP_BLENDED	EX	Pa	xy	None	tropopause pressure based on blended estimate
TROPT	EX	K	xy	None	tropopause temperature using blended TROPP estimate
TROPQ	EX	kg/kg	xy	None	tropopause specific humidity using blended TROPP estimate
DELP	EX	Pa	xyz	Center	pressure thickness
U_CGRID	EX	m s^{-1}	xyz	Center	eastward wind on C-Grid
V_CGRID	EX	m s^{-1}	xyz	Center	northward wind on C-Grid
U_DGRID	EX	m s^{-1}	xyz	Center	eastward wind on native D-Grid
V_DGRID	EX	m s^{-1}	xyz	Center	northward wind on native D-Grid
TV	EX	K	xyz	Center	air virtual temperature
THV	EX	K/Pa^{κ}	xyz	Center	scaled virtual potential temperature
DDELPDTDYN	EX	Pa sec^{-1}	xyz	Center	tendency of pressure thickness due to dynamics
UKE	EX	J m^{-1} s^{-1}	xy	None	eastward flux of atmospheric kinetic energy
VKE	EX	J m^{-1} s^{-1}	xy	None	northward flux of atmospheric kinetic energy
UCPT	EX	J m^{-1} s^{-1}	xy	None	eastward flux of atmospheric enthalpy
VCPT	EX	J m^{-1} s^{-1}	xy	None	northward flux of atmospheric enthalpy
UPHI	EX	J m^{-1} s^{-1}	xy	None	eastward flux of atmospheric potential energy
VPHI	EX	J m^{-1} s^{-1}	xy	None	northward flux of atmospheric potential energy
UQV	EX	kg m^{-1} s^{-1}	xy	None	eastward flux of atmospheric water vapor
VQV	EX	kg m^{-1} s^{-1}	xy	None	northward flux of atmospheric water vapor
UQL	EX	kg m^{-1} s^{-1}	xy	None	eastward flux of atmospheric liquid water
VQL	EX	kg m^{-1} s^{-1}	xy	None	northward flux of atmospheric liquid water

Short Name	Type	Units	Dims	Vert Loc	Long name
UQI	EX	kg m^{-1} s^{-1}	xy	None	eastward flux of atmospheric ice
VQI	EX	kg m^{-1} s^{-1}	xy	None	northward flux of atmospheric ice
DKE	EX	W m^{-2}	xy	None	tendency of atmosphere kinetic energy content due to dynamics
DCPT	EX	W m^{-2}	xy	None	tendency of atmosphere dry energy content due to dynamics
DPET	EX	W m^{-2}	xy	None	tendency of atmosphere topographic potential energy due to dynamics
WRKT	EX	W m^{-2}	xy	None	work done by atmosphere at top
DQV	EX	kg m^{-2} s^{-1}	xy	None	tendency of atmosphere water vapor content due to dynamics
DQL	EX	kg m^{-2} s^{-1}	xy	None	tendency of atmosphere liquid water content due to dynamics
DQI	EX	kg m^{-2} s^{-1}	xy	None	tendency of atmosphere ice content due to dynamics
CNV	EX	W m^{-2}	xy	None	generation of atmosphere kinetic energy content
U850	EX	m s^{-1}	xy	None	eastward wind at 850 hPa
U500	EX	m s^{-1}	xy	None	eastward wind at 500 hPa
U250	EX	m s^{-1}	xy	None	eastward wind at 250 hPa
V850	EX	m s^{-1}	xy	None	northward wind at 850 hPa
V500	EX	m s^{-1}	xy	None	northward wind at 500 hPa
V250	EX	m s^{-1}	xy	None	northward wind at 250 hPa
T850	EX	K	xy	None	air temperature at 850 hPa
T500	EX	K	xy	None	air temperature at 500 hPa
T250	EX	K	xy	None	air temperature at 250 hPa
Q850	EX	kg kg^{-1}	xy	None	specific humidity at 850 hPa
Q500	EX	kg kg^{-1}	xy	None	specific humidity at 500 hPa
Q250	EX	kg kg^{-1}	xy	None	specific humidity at 250 hPa
H850	EX	m	xy	None	height at 850 hPa
H500	EX	m	xy	None	height at 500 hPa
H250	EX	m	xy	None	height at 250 hPa
OMEGA500	EX	Pa s^{-1}	xy	None	omega at 500 hPa
U50M	EX	m s^{-1}	xy	None	eastward wind at 50 meters
V50M	EX	m s^{-1}	xy	None	northward wind at 50 meters
AREA	EX	m^2	xy	None	agrid cell area
AK	IN	Pa	z	Edge	hybrid sigma pressure a
BK	IN	1	z	Edge	hybrid sigma pressure b
U	IN	m s^{-1}	xyz	Center	eastward wind
V	IN	m s^{-1}	xyz	Center	northward wind

Short Name	Type	Units	Dims	Vert Loc	Long name
PT	IN	K $Pa^{-\kappa}$	xyz	Center	scaled potential temperature
PE	IN	Pa	xyz	Edge	air pressure
PKZ	IN	Pa^{κ}	xyz	Center	pressure to kappa

1.2 Finalize

DESCRIPTION:
Writes restarts and cleans-up through MAPL_GenericFinalize and deallocates memory from the Private Internal state. INTERFACE:

```
subroutine Finalize(gc, import, export, clock, rc)
   use dynamics_vars, only : dynamics_clean
```

ARGUMENTS:

```
type (ESMF_GridComp), intent(inout) :: gc
type (ESMF_State),    intent(inout) :: import
type (ESMF_State),    intent(inout) :: export
type (ESMF_Clock),    intent(inout) :: clock
integer, optional,    intent(  out) :: rc
```

1.3 Coldstart

DESCRIPTION:
Routine to coldstart from an isothermal state of rest. The temperature can be specified in the config, otherwise it is 300K. The surface pressure is assumed to be 1000 hPa. INTERFACE:

```
subroutine Coldstart(gc, import, export, clock, rc)
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: gc
type(ESMF_State),    intent(inout) :: import
type(ESMF_State),    intent(inout) :: export
type (ESMF_Clock),   intent(in)    :: clock
integer, intent(out), optional     :: rc
```

RESOURCES:

Name	Description	Units	Default
'T0:'	Value of isothermal temperature on coldstart	K	300.

2 Module BC_GridCompMod — BC Grid Component Class

INTERFACE:

```
module  BCGridCompMod
```

USES:

```
USE ESMF
USE MAPL_Mod

use Chem_Mod              ! Chemistry Base Class
use Chem_StateMod         ! Chemistry State
use Chem_ConstMod, only: grav, von_karman, cpd, &
                         undefval  = > undef         ! Constants !
use Chem_UtilMod          ! I/O
use Chem_MieMod           ! Aerosol LU Tables, calculator
use m_inpak90             ! Resource file management
use m_die, only: die
use Chem_SettlingMod      ! Settling
use DryDepositionMod      ! Dry Deposition
use WetRemovalMod         ! Large-scale Wet Removal
use ConvectionMod         ! Offline convective mixing/scavenging
```

PUBLIC TYPES:

```
PRIVATE
PUBLIC  BC_GridComp       ! The BC object
```

PUBLIC MEMBER FUNCTIONS:

```
PUBLIC  BC_GridCompInitialize
PUBLIC  BC_GridCompRun
PUBLIC  BC_GridCompFinalize
```

DESCRIPTION:
This module implements the (pre-ESMF) BC Grid Component. REVISION HISTORY:

```
16Sep2003 da Silva  First crack.
```

2.1 BC_GridCompInitialize — Initialize BC_GridComp

INTERFACE:

```
subroutine BC_GridCompInitialize ( gcBC, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

INPUT PARAMETERS:

```
type(Chem_Bundle), intent(inout) :: w_c     ! Chemical tracer fields
integer, intent(in) :: nymd, nhms           ! time
real, intent(in) :: cdt                     ! chemistry timestep (secs)
```

OUTPUT PARAMETERS:

```
type(BC_GridComp), intent(inout) :: gcBC     ! Grid Component
type(ESMF_State), intent(inout)  :: impChem  ! Import State
type(ESMF_State), intent(inout)  :: expChem  ! Export State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Initializes the BC Grid Component. It primarily sets the import state for each active constituent package. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

2.2 BC_GridCompRun — The Chem Driver

INTERFACE:

```
subroutine BC_GridCompRun ( gcBC, w_c, impChem, expChem, &
                            nymd, nhms, cdt, rc )
```

USES:

INPUT/OUTPUT PARAMETERS:

```
type(BC_GridComp), intent(inout) :: gcBC   ! Grid Component
type(Chem_Bundle), intent(inout) :: w_c      ! Chemical tracer fields
```

INPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: impChem ! Import State
integer, intent(in) :: nymd, nhms          ! time
real, intent(in) :: cdt                    ! chemistry timestep (secs)
```

OUTPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: expChem   ! Export State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This routine implements the so-called BC Driver. That is, adds chemical tendencies to each of the constituents, Note: water wapor, the first constituent is not considered a chemical constituents. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

2.3 BC_Emission - Adds Black Carbon emission for one timestep

We have emissions from 4 sources, which are distributed differently in the vertical 1) biomass burning - uniformly mixed in PBL 2) biofuel sources - emitted into lowest 100 m 3) anthropogenic l1 - emitted into lowest 100 m 4) anthropogenic l2 - emitted into 100 - 500 m levels
INTERFACE:

```
subroutine BC_Emission ( i1, i2, j1, j2, km, nbins, cdt, gcBC, w_c, &
                         pblh, tmpu, rhoa, BC_emis, &
                         BC_emisAN, BC_emisBB, BC_emisBF, rc )
```

USES:

INPUT PARAMETERS:

```
integer, intent(in) :: i1, i2, j1, j2, km, nbins
real, intent(in)    :: cdt
type(BC_GridComp), intent(in)    :: gcBC       ! BC Grid Component
real, pointer, dimension(:,:)    :: pblh
real, pointer, dimension(:,:,:)  :: tmpu
real, pointer, dimension(:,:,:)  :: rhoa
```

OUTPUT PARAMETERS:

```
type(Chem_Bundle), intent(inout) :: w_c         ! Chemical tracer fields
type(Chem_Array), intent(inout)  :: BC_emis(nbins) ! BC emissions, kg/m2/s
type(Chem_Array), intent(inout)  :: BC_emisAN      ! BC emissions, kg/m2/s
type(Chem_Array), intent(inout)  :: BC_emisBB      ! BC emissions, kg/m2/s
type(Chem_Array), intent(inout)  :: BC_emisBF      ! BC emissions, kg/m2/s
integer, intent(out)             :: rc          ! Error return code:
                                                !  0 - all is well
                                                !  1 -
character(len = *), parameter :: myname  =  'BC_Emission'
```

DESCRIPTION:
Updates the BC concentration with emissions every timestep REVISION HISTORY:

```
06Nov2003, Colarco
Based on Ginoux
```

2.4 BC_Compute_Diags - Calculate dust 2D diagnostics

INTERFACE:

```
subroutine BC_Compute_Diags ( i1, i2, j1, j2, km, nbins, gcBC, w_c, tmpu, rhoa, u, v, &
                              sfcmass, colmass, mass, exttau, scatau, &
                              conc, extcoef, scacoef, angstrom, fluxu, fluxv, rc )
```

USES:

INPUT PARAMETERS:

```
integer, intent(in) :: i1, i2, j1, j2, km, nbins
type(BC_GridComp), intent(inout):: gcBC     ! BC Grid Component
type(Chem_Bundle), intent(in)   :: w_c      ! Chem Bundle
real, pointer, dimension(:,:,:) :: tmpu     ! temperature [K]
real, pointer, dimension(:,:,:) :: rhoa     ! air density [kg m-3]
real, pointer, dimension(:,:,:) :: u        ! east-west wind [m s-1]
real, pointer, dimension(:,:,:) :: v        ! north-south wind [m s-1]
```

OUTPUT PARAMETERS:

```
type(Chem_Array), intent(inout)  :: sfcmass  ! sfc mass concentration kg/m3
type(Chem_Array), intent(inout)  :: colmass  ! col mass density kg/m2
type(Chem_Array), intent(inout)  :: mass     ! 3d mass mixing ratio kg/kg
type(Chem_Array), intent(inout)  :: exttau   ! ext. AOT at 550 nm
type(Chem_Array), intent(inout)  :: scatau   ! sct. AOT at 550 nm
type(Chem_Array), intent(inout)  :: conc     ! 3d mass concentration, kg/m3
type(Chem_Array), intent(inout)  :: extcoef  ! 3d ext. coefficient, 1/m
type(Chem_Array), intent(inout)  :: scacoef  ! 3d scat.coefficient, 1/m
type(Chem_Array), intent(inout)  :: angstrom ! 470-870 nm Angstrom parameter
type(Chem_Array), intent(inout)  :: fluxu    ! Column mass flux in x direction
type(Chem_Array), intent(inout)  :: fluxv    ! Column mass flux in y direction
integer, intent(out)             :: rc       ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Calculates some simple 2d diagnostics from the BC fields Surface concentration (dry) Column mass load (dry) Extinction aot 550 (wet) Scattering aot 550 (wet) For the moment, this is hardwired. REVISION HISTORY:

```
16APR2004, Colarco
```

2.5 BC_GridCompFinalize — The Chem Driver

INTERFACE:

```
subroutine BC_GridCompFinalize ( gcBC, w_c, impChem, expChem, &
                                 nymd, nhms, cdt, rc )
```

USES:

INPUT/OUTPUT PARAMETERS:

```
type(BC_GridComp), intent(inout) :: gcBC   ! Grid Component
```

INPUT PARAMETERS:

```
type(Chem_Bundle), intent(in)  :: w_c      ! Chemical tracer fields
integer, intent(in) :: nymd, nhms          ! time
real,    intent(in) :: cdt                 ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: impChem   ! Import State
type(ESMF_State), intent(inout) :: expChem   ! Import State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This routine finalizes this Grid Component. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

3 Module CFC_GridCompMod — CFC Grid Component Class

INTERFACE:

```
MODULE  CFCGridCompMod
```

USES:

```
USE ESMF
USE MAPL_Mod
USE Chem_Mod       ! Chemistry Base Class
USE Chem_StateMod      ! Chemistry State
USE Chem_ConstMod, ONLY: grav
USE Chem_UtilMod      ! I/O
USE m_inpak90      ! Resource file management

USE ESMF_CFIOFileMOD
USE MAPL_CFIOMOD

IMPLICIT NONE
```

PUBLIC TYPES:

```
PRIVATE
PUBLIC  CFC_GridComp       ! The CFC object
```

PUBLIC MEMBER FUNCTIONS:

```
PUBLIC  CFC_GridCompInitialize
PUBLIC  CFC_GridCompRun
PUBLIC  CFC_GridCompFinalize
```

DESCRIPTION:
This module implements the CFC Grid Component. REVISION HISTORY:

```
16Sep2003 da Silva  First crack.
01Aug2006 da Silva  Extensions for GEOS-5.
 1Jan2008  Nielsen  CFC-12 configuration for ARCTAS.
 8Feb2008  Nielsen  Standard configuration call(s) from AeroChem.
```

3.1 CFC_GridCompInitialize — Initialize CFC_GridComp

INTERFACE:

```
SUBROUTINE CFC_GridCompInitialize( gcCFC, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), intent(in) :: w_c        ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms        ! time
REAL,    INTENT(IN) :: cdt        ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(CFC_GridComp), INTENT(INOUT) :: gcCFC   ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem  ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem  ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Initializes the CFC Grid Component. It primarily sets the import state for each active constituent package. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
31May2005  Nielsen  Mods for 7 CO bins, 5 region masks
04Nov2005     Bian  CO tagged to 4 regions
                    (global, North America, South America, and Africa)
                    for CR-AVE
```

```
12Feb2005  Nielsen  8 regions for INTEX-B 2006
 1Jan2008  Nielsen  CFC-12 configuration for ARCTAS
 1Nov2012  Nielsen  Accomodate cubed sphere for GEOS-5 Ganymed releases
```

INTERFACE:

```
SUBROUTINE readPhotTables(fileName, rc)
```

USES:

```
IMPLICIT NONE
```

INPUT PARAMETERS:

```
CHARACTER(LEN = *), INTENT(IN) :: fileName
```

OUTPUT PARAMETERS:

```
INTEGER, INTENT(OUT) :: rc
```

DESCRIPTION:
Read tables for photolysis in StratChem ... from a NetCDF file
Restrictions: ASSERT that the number of pressure layers in the dataset equals km. REVISION HISTORY:

```
Nielsen     11 May 2012: First crack.
```

3.2 CFC_GridCompRun — The CFC Driver

INTERFACE:

```
SUBROUTINE CFC_GridCompRun( gcCFC, w_c, impChem, expChem, nymd, nhms, &
                            cdt, rc)
```

USES:

```
IMPLICIT NONE
```

INPUT/OUTPUT PARAMETERS:

```
TYPE(CFC_GridComp), INTENT(INOUT) :: gcCFC   ! Grid Component
TYPE(Chem_Bundle), INTENT(INOUT) :: w_c ! Chemical tracer fields
```

INPUT PARAMETERS:

```
TYPE(ESMF_State), INTENT(INOUT) :: impChem   ! Import State
INTEGER, INTENT(IN) :: nymd, nhms         ! time
REAL,    INTENT(IN) :: cdt         ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(ESMF_State), INTENT(INOUT) :: expChem   ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

3.3 CFC_GridCompFinalize

INTERFACE:

```
SUBROUTINE CFC_GridCompFinalize( gcCFC, w_c, impChem, expChem, &
                                 nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT/OUTPUT PARAMETERS:

```
TYPE(CFC_GridComp), INTENT(INOUT) :: gcCFC ! Grid Component
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), INTENT(IN)  :: w_c      ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms       ! time
REAL,    INTENT(IN) :: cdt       ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(ESMF_State), INTENT(INOUT) :: impChem ! Import State
TYPE(ESMF_State), INTENT(INOUT) :: expChem ! Import State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This routine finalizes this Grid Component. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

4 Module CH4_GridCompMod — CH4 Grid Component Class

INTERFACE:

```
MODULE  CH4GridCompMod
```

USES:

```
USE ESMF
USE MAPL_Mod
USE Chem_Mod       ! Chemistry Base Class
USE Chem_StateMod      ! Chemistry State
USE Chem_UtilMod      ! I/O
USE m_inpak90      ! Resource file management
USE m_chars, ONLY: lowercase, uppercase

IMPLICIT NONE
```

PUBLIC TYPES:

```
PRIVATE
PUBLIC  CH4_GridComp       ! Multiple instance CH4 object
PUBLIC  CH4_GridComp1      ! Single instance CH4 object
```

PUBLIC MEMBER FUNCTIONS:

```
PUBLIC  CH4_GridCompInitialize
PUBLIC  CH4_GridCompRun
PUBLIC  CH4_GridCompFinalize
```

DESCRIPTION:
This module implements the CH4 Grid Component. REVISION HISTORY:

```
24 Jun 2010 Nielsen:  First crack.
25 Oct 2012 Nielsen:  Added photolysis.
```

4.1 CH4_GridCompInitialize — Initialize CH4_GridComp

INTERFACE:

```
subroutine CH4_GridCompInitialize ( gcCH4, w_c, impChem, expChem, &
                                    nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), intent(in) :: w_c        ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms        ! time
REAL,    INTENT(IN) :: cdt        ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(CH4_GridComp), INTENT(INOUT) :: gcCH4   ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem  ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem  ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Initializes the CH4 Grid Component. Multiple instance version. REVISION HISTORY:

```
24 Jun 2010 Nielsen:  First crack.
25 Oct 2012 Nielsen:  Added photolysis.
```

4.2 CH4_GridCompRun — Run CH4_GridComp

INTERFACE:

```
subroutine CH4_GridCompRun ( gcCH4, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), intent(in) :: w_c        ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms        ! time
REAL,    INTENT(IN) :: cdt        ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(CH4_GridComp), INTENT(INOUT) :: gcCH4   ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem  ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem  ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Runs the CH4 Grid Component. Multiple instance version. REVISION HISTORY:

```
24 Jun 2010 Nielsen:  First crack.
25 Oct 2012 Nielsen:  Added photolysis.
```

4.3 CH4_GridCompFinalize — Initialize CH4_GridComp

INTERFACE:

```
subroutine CH4_GridCompFinalize ( gcCH4, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), intent(in) :: w_c        ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms        ! time
REAL,    INTENT(IN) :: cdt        ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(CH4_GridComp), INTENT(INOUT) :: gcCH4   ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem  ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem  ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Finalizes the CH4 Grid Component. Multiple instance version. REVISION HISTORY:

```
27Feb2008  da Silva  Introduced multiple instances
```

4.4 CH4_GridCompInitialize — Initialize CH4_GridComp

INTERFACE:

```
subroutine CH4_GridCompInitialize1_ ( gcCH4, w_c, impChem, expChem, &
                                       nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), intent(in) :: w_c        ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms        ! time
REAL,    INTENT(IN) :: cdt        ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(CH4_GridComp1), INTENT(INOUT) :: gcCH4   ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem  ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem  ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Initializes the CH4 Grid Component. It primarily sets the import state for each active constituent package. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
31May2005  Nielsen  Mods for 7 CH4 bins, 5 region masks
04Nov2005     Bian  CO tagged to 4 regions
                    (global, North America, South America, and Africa)
                    for CR-AVE
25Oct2012  Nielsen  Added photolysis.
```

4.5 CH4_GridCompRun

INTERFACE:

```
SUBROUTINE CH4_GridCompRun1_ ( gcCH4, w_c, impChem, expChem, &
                               nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT/OUTPUT PARAMETERS:

```
TYPE(CH4_GridComp1), INTENT(INOUT) :: gcCH4   ! Grid Component
TYPE(Chem_Bundle), INTENT(INOUT) :: w_c ! Chemical tracer fields
```

INPUT PARAMETERS:

```
TYPE(ESMF_State), INTENT(inout) :: impChem    ! Import State
INTEGER, INTENT(IN) :: nymd, nhms       ! time
REAL,    INTENT(IN) :: cdt       ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(ESMF_State), intent(inout) :: expChem     ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This routine implements the CH4 Driver for GOCART. REVISION HISTORY:

```
24 Jun 2010 Nielsen:  First crack.
25 Oct 2012 Nielsen:  Added photolysis.
```

4.6 CH4_GridCompFinalize — The Chem Driver

INTERFACE:

```
SUBROUTINE CH4_GridCompFinalize1_ ( gcCH4, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT/OUTPUT PARAMETERS:

```
TYPE(CH4_GridComp1), INTENT(INOUT) :: gcCH4   ! Grid Component
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), INTENT(IN)  :: w_c      ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms       ! time
REAL,    INTENT(IN) :: cdt       ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(ESMF_State), INTENT(INOUT) :: impChem ! Import State
TYPE(ESMF_State), INTENT(INOUT) :: expChem ! Import State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This routine finalizes this Grid Component. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

4.7 CH4_SingleInstance_ — Runs single instance of method

INTERFACE:

```
subroutine CH4_SingleInstance_ ( Method_, instance, &
                                 gcCH4, w_c, impChem, expChem, &
                                 nymd, nhms, cdt, rc )
```

USES:

```
Use CH4_GridCompMod
Use ESMF
Use MAPL_Mod
Use Chem_Mod

IMPLICIT NONE
```

INPUT PARAMETERS:

```
Input "function pointer"
-----------------------
interface
  subroutine Method_ (gc, w, imp, exp, ymd, hms, dt, rcode )
    Use CH4_GridCompMod
```

```
    Use ESMF
    Use MAPL_Mod
    Use Chem_Mod
    type(CH4_GridComp1),  intent(inout)  :: gc
    type(Chem_Bundle),   intent(in)     :: w
    type(ESMF_State),    intent(inout)  :: imp
    type(ESMF_State),    intent(inout)  :: exp
    integer,             intent(in)     :: ymd, hms
    real,                intent(in)     :: dt
    integer,             intent(out)    :: rcode
  end subroutine Method_
end interface

integer, intent(in)           :: instance   ! instance number

TYPE(Chem_Bundle), intent(inout) :: w_c     ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms        ! time
REAL,    INTENT(IN) :: cdt        ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(CH4_GridComp1), INTENT(INOUT) :: gcCH4    ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem  ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem  ! Export State
INTEGER, INTENT(OUT) ::  rc                 ! Error return code:
                                            !  0 - all is well
                                            !  1 -
```

DESCRIPTION:
Finalizes the CH4 Grid Component. Multiple instance version. REVISION HISTORY:

```
27Feb2008  da Silva  Introduced multiple instances
```

5 Module CO2_GridCompMod — CO2 Grid Component Class

INTERFACE:

```
module  CO2GridCompMod
```

USES:

```
USE ESMF
USE MAPL_Mod
```

```
use Chem_Mod              ! Chemistry Base Class
use Chem_StateMod         ! Chemistry State
use Chem_ConstMod, only: grav
use Chem_UtilMod
use m_inpak90             ! Resource file management
```

PUBLIC TYPES:

```
PRIVATE
PUBLIC  CO2_GridComp       ! The CO2 object

!PUBLIIC MEMBER FUNCTIONS:

PUBLIC  CO2_GridCompInitialize
PUBLIC  CO2_GridCompRun
PUBLIC  CO2_GridCompFinalize
```

DESCRIPTION:
This module implements the (pre-ESMF) CO2 Grid Component. REVISION HISTORY:

```
16Sep2003 da Silva  First crack.
24Oct2005     Bian  tag CO2 to 4 regions
                    (total, north america, south america, africa)
19Dec2005 da Silva  Activated 3D diags for output
26Nov2010  Nielsen  Simplified PBL partitioning for biomass burning emissions
```

5.1 CO2_GridCompInitialize — Initialize CO2_GridComp

INTERFACE:

```
subroutine CO2_GridCompInitialize ( gcCO2, w_c, impChem, expChem, &
                                    nymd, nhms, cdt, rc )
```

USES:

INPUT PARAMETERS:

```
type(Chem_Bundle), intent(in) :: w_c        ! Chemical tracer fields
integer, intent(in) :: nymd, nhms           ! time
real,    intent(in) :: cdt                  ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
type(CO2_GridComp), intent(inout) :: gcCO2   ! Grid Component
type(ESMF_State), intent(inout)  :: impChem  ! Import State
type(ESMF_State), intent(inout)  :: expChem  ! Export State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Initializes the CO2 Grid Component. It primarily sets the import state for each active constituent package. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
24OCT2005     Bian  Mods for 5 tagged CO2
                    (total, fossil fuel, ecosystem, oceanic, and biomass)
25OCT2005     Bian  Mods for 5 regions
```

5.2 CO2_GridCompRun — The Chem Driver

INTERFACE:

```
subroutine CO2_GridCompRun ( gcCO2, w_c, impChem, expChem, &
                             nymd, nhms, cdt, rc )
```

USES:

INPUT/OUTPUT PARAMETERS:

```
type(CO2_GridComp), intent(inout) :: gcCO2   ! Grid Component
type(Chem_Bundle), intent(inout) :: w_c      ! Chemical tracer fields
```

INPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: impChem    ! Import State
integer, intent(in) :: nymd, nhms           ! time
real,    intent(in) :: cdt                  ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: expChem     ! Export State
integer, intent(out) ::  rc                   ! Error return code:
                                              !  0 - all is well
                                              !  1 -
```

DESCRIPTION:
This routine implements the so-called CO2 Driver. That is, adds chemical tendencies to each of the constituents, Note: water wapor, the first constituent is not considered a chemical constituents. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
24OCT2005     Bian  Mods for 5 tagged CO2
                    (total, fossil fuel, ecosystem, oceanic, and biomass)
25OCT2005     Bian  Mods for 5 regions

Mask  Region
----  -------------
  1   North America
  2   Mexico
  3   Europe
  4   Asia
  5   Africa
```

5.3 CO2_Emission - Adds emissions for CO2 for one timestep

We have emissions from 4 sources, which are distributed differently in the vertical 1) fossil fuel - emitted at surface 2) ecosystem - fluxes at surface 3) oceanic - fluxes at surface 4) biomass burning - uniformly mixed in PBL
DESCRIPTION:
Updates the CO2 concentration with emissions every timestep REVISION HISTORY:

```
24Oct2005, Bian
26Nov2010, Nielsen Simplified PBL partitioning for biomass burning emissions
```

INTERFACE:

5.4 CO2_GridCompFinalize — The Chem Driver

INTERFACE:

```
subroutine CO2_GridCompFinalize ( gcCO2, w_c, impChem, expChem, &
                                  nymd, nhms, cdt, rc )
```

USES:

INPUT/OUTPUT PARAMETERS:

```
type(CO2_GridComp), intent(inout) :: gcCO2   ! Grid Component
```

INPUT PARAMETERS:

```
type(Chem_Bundle), intent(in)  :: w_c      ! Chemical tracer fields
integer, intent(in) :: nymd, nhms          ! time
real,    intent(in) :: cdt                 ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: impChem   ! Import State
type(ESMF_State), intent(inout) :: expChem   ! Import State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This routine finalizes this Grid Component. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

6 Module CO_GridCompMod — CO Grid Component Class

INTERFACE:

```
MODULE  COGridCompMod
```

USES:

```
USE ESMF
USE MAPL_Mod

USE Chem_Mod                        ! Chemistry Base Class
USE Chem_StateMod                   ! Chemistry State
USE Chem_ConstMod, only: grav
USE Chem_UtilMod                    ! I/O

USE m_inpak90                       ! Resource file management
USE m_die, ONLY: die
USE m_chars, ONLY: lowercase

IMPLICIT NONE
```

PUBLIC TYPES:

```
PRIVATE
PUBLIC  CO_GridComp                 ! Multiple instance CO object
PUBLIC  CO_GridComp1                ! Single instance CO object
```

PUBLIC MEMBER FUNCTIONS:

```
PUBLIC  CO_GridCompInitialize
PUBLIC  CO_GridCompRun
PUBLIC  CO_GridCompFinalize
```

DESCRIPTION:
This module implements the (pre-ESMF) CO Grid Component. REVISION HISTORY:

```
16Sep2003 da Silva  First crack.
31May2005  Nielsen  Mods for 7 CO bins, 5 region masks
31May2005 da Silva  Seperate file for biomass emissions; option for
                     daily templatable files
31May2005 da Silva  Moved reading of region mask to init, specified
                    fixed time.
17Oct2005     Bian  add biogenic emission and CH4 oxidation, two options
                    for updating emissions
19dec2005 da Silva  Activated 3D diags for output
14Apr2006     Bian  Add CO tagged to fossil fuel, biofuel, biomass burning
                    and biogenic
  Oct2006     Bian  Evaluate total and tagged CO performace in GEOS4 system
                    with emissions and oxident fields described in
                    Bian et al., [2007]. The observations included GMD ground
                    surface and aircraft measurements, TRACE-P aircraft
```

```
                      measurements, and satellite MOPITT and AIRS retrieves.
  01Aug2006 da Silva  Extensions for GEOS-5.
  10Mar2008 da Silva  Multiple instances for ARCTAS.
  18Mar2011  Nielsen  Simplified PBL partitioning for biomass burning emissions
```

6.1 CO_GridCompInitialize — Initialize CO_GridComp

INTERFACE:

```
   subroutine CO_GridCompInitialize ( gcCO, w_c, impChem, expChem, &
                                      nymd, nhms, cdt, rc )
```

USES:

```
  IMPLICIT NONE
```

INPUT PARAMETERS:

```
   TYPE(Chem_Bundle), intent(in) :: w_c          ! Chemical tracer fields
   INTEGER, INTENT(IN) :: nymd, nhms             ! time
   REAL,    INTENT(IN) :: cdt                    ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
   TYPE(CO_GridComp), INTENT(INOUT) :: gcCO      ! Grid Component
   TYPE(ESMF_State), INTENT(INOUT)  :: impChem   ! Import State
   TYPE(ESMF_State), INTENT(INOUT)  :: expChem   ! Export State
   INTEGER, INTENT(OUT) ::  rc                   ! Error return code:
                                                 !  0 - all is well
                                                 !  1 -
```

DESCRIPTION:
Initializes the CO Grid Component. Multiple instance version. REVISION HISTORY:

```
   27Feb2008  da Silva  Introduced multiple instances
```

6.2 CO_GridCompRun — Run CO_GridComp

INTERFACE:

```
subroutine CO_GridCompRun ( gcCO, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), intent(in) :: w_c          ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms             ! time
REAL,    INTENT(IN) :: cdt                    ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(CO_GridComp), INTENT(INOUT) :: gcCO      ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem   ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem   ! Export State
INTEGER, INTENT(OUT) ::  rc                   ! Error return code:
                                              !  0 - all is well
                                              !  1 -
```

DESCRIPTION:
Runs the CO Grid Component. Multiple instance version. REVISION HISTORY:

```
27Feb2008  da Silva  Introduced multiple instances
```

6.3 CO_GridCompFinalize — Initialize CO_GridComp

INTERFACE:

```
subroutine CO_GridCompFinalize ( gcCO, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), intent(in) :: w_c          ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms             ! time
REAL,    INTENT(IN) :: cdt                    ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(CO_GridComp), INTENT(INOUT) :: gcCO      ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem   ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem   ! Export State
INTEGER, INTENT(OUT) ::  rc                   ! Error return code:
                                              !  0 - all is well
                                              !  1 -
```

DESCRIPTION:
Finalizes the CO Grid Component. Multiple instance version. REVISION HISTORY:

```
27Feb2008  da Silva  Introduced multiple instances
```

6.4 CO_GridCompInitialize — Initialize CO_GridComp

INTERFACE:

```
subroutine CO_GridCompInitialize1_ ( gcCO, w_c, impChem, expChem, &
                                     nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), intent(in) :: w_c          ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms             ! time
REAL,    INTENT(IN) :: cdt                    ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(CO_GridComp1), INTENT(INOUT) :: gcCO     ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem   ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem   ! Export State
INTEGER, INTENT(OUT) ::  rc                   ! Error return code:
                                              !  0 - all is well
                                              !  1 -
```

DESCRIPTION:
Initializes the CO Grid Component. It primarily sets the import state for each active constituent package. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
31May2005  Nielsen  Mods for 7 CO bins, 5 region masks
04Nov2005     Bian  CO tagged to 4 regions
                    (global, North America, South America, and Africa)
                    for CR-AVE
```

6.5 CO_GridCompRun — The Chem Driver

INTERFACE:

```
SUBROUTINE CO_GridCompRun1_ ( gcCO, w_c, impChem, expChem, &
                              nymd, nhms, cdt, rc )
```

```
#define SFLUX_PBL
```

USES:

```
IMPLICIT NONE
```

INPUT/OUTPUT PARAMETERS:

```
TYPE(CO_GridComp1), INTENT(INOUT) :: gcCO     ! Grid Component
TYPE(Chem_Bundle), INTENT(INOUT) :: w_c       ! Chemical tracer fields
```

INPUT PARAMETERS:

```
TYPE(ESMF_State), INTENT(inout) :: impChem    ! Import State
INTEGER, INTENT(IN) :: nymd, nhms             ! time
REAL,    INTENT(IN) :: cdt                    ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(ESMF_State), intent(inout) :: expChem    ! Export State
INTEGER, INTENT(OUT) ::  rc                   ! Error return code:
                                              !  0 - all is well
                                              !  1 -
```

DESCRIPTION:
This routine implements the CO Driver for INTEX. That is, adds chemical tendencies to each of the constituents, Note: water wapor, the first constituent is not considered a chemical constituents. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
31May2005  Nielsen  Mods for 7 tags, 5 regions
04Nov2005     Bian  CO tagged to 4 regions
13Apr2005     Bian  CO tagged to emissions
```

DESCRIPTION:
Updates the CO concentration with emissions every timestep

6.6 CO_Emission - Adds emissions for CO for one timestep

We have emissions from 4 sources, which are distributed differently in the vertical 1) fossil fuel - emitted at surface 2) biofuel sources - emitted at surface 3) biomass burning - uniformly mixed in PBL 4) biogenic - emitted at surface include: isoprene, converting factor 0.15 terpene, converting factor 0.2 nvoc, converting factor 0.2REVISION HISTORY:

```
17Oct2005, Bian!
14Apr2006, Bian: Add indirect NMHC from FF (0.20), BF (0.19), BB (0.11)
                 Add seasonality for FF
                 Modify FF & BF over Asia region (1.39) for Streets' data
18Mar2011, Nielsen: Simplified PBL partitioning for biomass burning emissions
```

INTERFACE:

6.7 CO_GridCompFinalize — The Chem Driver

INTERFACE:

```
SUBROUTINE CO_GridCompFinalize1_ ( gcCO, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT/OUTPUT PARAMETERS:

```
TYPE(CO_GridComp1), INTENT(INOUT) :: gcCO     ! Grid Component
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), INTENT(IN)  :: w_c         ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms             ! time
REAL,    INTENT(IN) :: cdt                    ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(ESMF_State), INTENT(INOUT) :: impChem    ! Import State
TYPE(ESMF_State), INTENT(INOUT) :: expChem    ! Import State
INTEGER, INTENT(OUT) ::  rc                   ! Error return code:
                                              !  0 - all is well
                                              !  1 -
```

DESCRIPTION:
This routine finalizes this Grid Component. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

6.8 CO_SingleInstance_ — Runs single instance of method

INTERFACE:

```
subroutine CO_SingleInstance_ ( Method_, instance, &
                                gcCO, w_c, impChem, expChem, &
                                nymd, nhms, cdt, rc )
```

USES:

```
Use CO_GridCompMod
Use ESMF
Use MAPL_Mod
Use Chem_Mod

IMPLICIT NONE
```

INPUT PARAMETERS:

```
Input "function pointer"
-----------------------
interface
  subroutine Method_ (gc, w, imp, exp, ymd, hms, dt, rcode )
    Use CO_GridCompMod
    Use ESMF
    Use MAPL_Mod
    Use Chem_Mod
    type(CO_GridComp1),  intent(inout)  :: gc
    type(Chem_Bundle),   intent(in)     :: w
    type(ESMF_State),    intent(inout)  :: imp
    type(ESMF_State),    intent(inout)  :: exp
    integer,             intent(in)     :: ymd, hms
    real,                intent(in)     :: dt
    integer,             intent(out)    :: rcode
  end subroutine Method_
end interface

integer, intent(in)           :: instance     ! instance number

TYPE(Chem_Bundle), intent(inout) :: w_c       ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms             ! time
REAL,    INTENT(IN) :: cdt                    ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(CO_GridComp1), INTENT(INOUT) :: gcCO     ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem   ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem   ! Export State
INTEGER, INTENT(OUT) ::  rc                   ! Error return code:
                                              !  0 - all is well
                                              !  1 -
```

DESCRIPTION:
Finalizes the CO Grid Component. Multiple instance version. REVISION HISTORY:

```
27Feb2008  da Silva  Introduced multiple instances
```

7 Module DU_GridCompMod — DU Grid Component Class

INTERFACE:

```
module  DUGridCompMod
```

USES:

```
USE ESMF
USE MAPL_Mod

use Chem_Mod              ! Chemistry Base Class
use Chem_StateMod         ! Chemistry State
use Chem_ConstMod, only: grav, von_karman, cpd, &
                         undefval  = > undef         ! Constants !
use Chem_UtilMod          ! I/O
use Chem_MieMod           ! Aerosol LU Tables, calculator
use m_inpak90             ! Resource file management
use m_die, only: die
use m_mpout
use DustEmissionMod       ! Emissions
use Chem_SettlingMod      ! Settling
use DryDepositionMod      ! Dry Deposition
use WetRemovalMod         ! Large-scale Wet Removal
use ConvectionMod         ! Offline convective mixing/scavenging
```

PUBLIC TYPES:

```
PRIVATE
PUBLIC  DU_GridComp       ! The DU object
```

PUBLIC MEMBER FUNCTIONS:

```
PUBLIC  DU_GridCompInitialize
PUBLIC  DU_GridCompRun
PUBLIC  DU_GridCompFinalize
```

DESCRIPTION:
This module implements the (pre-ESMF) DU Grid Component. REVISION HISTORY:

```
16Sep2003 da Silva  First crack.
16Aug2005 da Silva  Passed ESMF grid to MPread().
```

7.1 DU_GridCompInitialize — Initialize DU_GridComp

INTERFACE:

```
subroutine DU_GridCompInitialize ( gcDU, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

INPUT PARAMETERS:

```
type(Chem_Bundle), intent(inout) :: w_c        ! Chemical tracer fields
integer, intent(in) :: nymd, nhms            ! time
real, intent(in) :: cdt                      ! chemistry timestep (secs)
```

OUTPUT PARAMETERS:

```
type(DU_GridComp), intent(inout) :: gcDU   ! Grid Component
type(ESMF_State), intent(inout)  :: impChem  ! Import State
type(ESMF_State), intent(inout)  :: expChem  ! Export State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Initializes the DU Grid Component. It primarily sets the import state for each active constituent package. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

7.2 DU_GridCompRun — The Chem Driver

INTERFACE:

```
subroutine DU_GridCompRun ( gcDU, w_c, impChem, expChem, &
                            nymd, nhms, cdt, rc )
```

USES:

INPUT/OUTPUT PARAMETERS:

```
type(DU_GridComp), intent(inout) :: gcDU   ! Grid Component
type(Chem_Bundle), intent(inout) :: w_c    ! Chemical tracer fields
```

INPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: impChem    ! Import State
integer, intent(in) :: nymd, nhms           ! time
real, intent(in) :: cdt                     ! chemistry timestep (secs)
```

OUTPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: expChem   ! Export State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This routine implements the so-called DU Driver. That is, adds chemical tendencies to each of the constituents, Note: water wapor, the first constituent is not considered a chemical constituents. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

7.3 DU_Compute_Diags - Calculate dust 2D diagnostics

INTERFACE:

```
subroutine DU_Compute_Diags ( i1, i2, j1, j2, km, nbins, gcDU, w_c, tmpu, rhoa, &
                              u, v, sfcmass, colmass, mass, exttau, scatau,     &
                              sfcmass25, colmass25, mass25, exttau25, scatau25, &
                              aerindx, fluxu, fluxv, conc, extcoef, scacoef,    &
                              exttaufm, scataufm, angstrom, rc )
```

USES:

INPUT PARAMETERS:

```
integer, intent(in) :: i1, i2, j1, j2, km, nbins
type(DU_GridComp), intent(inout):: gcDU     ! DU Grid Component
type(Chem_Bundle), intent(in)   :: w_c      ! Chem Bundle
real, pointer, dimension(:,:,:) :: tmpu     ! temperature [K]
real, pointer, dimension(:,:,:) :: rhoa     ! air density [kg m-3]
```

```
real, pointer, dimension(:,:,:) :: u        ! east-west wind [m s-1]
real, pointer, dimension(:,:,:) :: v        ! north-south wind [m s-1]
```

OUTPUT PARAMETERS:

```
Total mass
type(Chem_Array), intent(inout)  :: sfcmass   ! sfc mass concentration kg/m3
type(Chem_Array), intent(inout)  :: colmass   ! col mass density kg/m2
type(Chem_Array), intent(inout)  :: mass      ! 3d mass mixing ratio kg/kg
Total optical properties
type(Chem_Array), intent(inout)  :: exttau    ! ext. AOT at 550 nm
type(Chem_Array), intent(inout)  :: scatau    ! sct. AOT at 550 nm
type(Chem_Array), intent(inout)  :: sfcmass25 ! sfc mass concentration kg/m3 (pm2.5)
type(Chem_Array), intent(inout)  :: colmass25 ! col mass density kg/m2 (pm2.5)
type(Chem_Array), intent(inout)  :: mass25    ! 3d mass mixing ratio kg/kg (pm2.5)
type(Chem_Array), intent(inout)  :: exttau25  ! ext. AOT at 550 nm (pm2.5)
type(Chem_Array), intent(inout)  :: scatau25  ! sct. AOT at 550 nm (pm2.5)
type(Chem_Array), intent(inout)  :: aerindx   ! TOMS UV AI
type(Chem_Array), intent(inout)  :: fluxu     ! Column mass flux in x direction
type(Chem_Array), intent(inout)  :: fluxv     ! Column mass flux in y direction
type(Chem_Array), intent(inout)  :: conc      ! 3d mass concentration, kg/m3
type(Chem_Array), intent(inout)  :: extcoef   ! 3d ext. coefficient, 1/m
type(Chem_Array), intent(inout)  :: scacoef   ! 3d scat.coefficient, 1/m
type(Chem_Array), intent(inout)  :: exttaufm  ! fine mode (sub-micron) ext. AOT at 550 nm
type(Chem_Array), intent(inout)  :: scataufm  ! fine mode (sub-micron) sct. AOT at 550 nm
type(Chem_Array), intent(inout)  :: angstrom  ! 470-870 nm Angstrom parameter
integer, intent(out)             :: rc        ! Error return code:
                                              !  0 - all is well
                                              !  1 -
```

DESCRIPTION:
Calculates some simple 2d diagnostics from the dust fields REVISION HISTORY:

```
16APR2004, Colarco
11MAR2010, Nowottnick
```

7.4 DU_Binwise_PM_Fractions - Calculate bin-wise PM fractions

INTERFACE:

```
subroutine DU_Binwise_PM_Fractions(fPM, rPM, r_low, r_up, nbins)
```

USES:

INPUT/OUTPUT PARAMETERS:

```
real, dimension(:), intent(inout) :: fPM     ! bin-wise PM fraction (r < rPM)
```

INPUT PARAMETERS:

```
real,    intent(in)              :: rPM     ! PM radius
integer, intent(in)              :: nbins   ! number of bins
real, dimension(:), intent(in)   :: r_low   ! bin radii - low bounds
real, dimension(:), intent(in)   :: r_up    ! bin radii - upper bounds
```

OUTPUT PARAMETERS:

7.5 DU_GridCompFinalize — The Chem Driver

INTERFACE:

```
subroutine DU_GridCompFinalize ( gcDU, w_c, impChem, expChem, &
                                 nymd, nhms, cdt, rc )
```

USES:

INPUT/OUTPUT PARAMETERS:

```
type(DU_GridComp), intent(inout) :: gcDU   ! Grid Component
```

INPUT PARAMETERS:

```
type(Chem_Bundle), intent(in)  :: w_c      ! Chemical tracer fields
integer, intent(in) :: nymd, nhms          ! time
real, intent(in) :: cdt                    ! chemistry timestep (secs)
```

OUTPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: impChem   ! Import State
type(ESMF_State), intent(inout) :: expChem   ! Export State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This routine finalizes this Grid Component. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

8 Module CreateInterpWeights_GridCompMod

USES:

```
use ESMF              ! ESMF base class
use MAPL_Mod          ! GEOS base class
```

PUBLIC MEMBER FUNCTIONS:

```
public  SetServices      ! Register component methods
```

8.1 SetServices

DESCRIPTION:
SetServices registers Initialize, Run, and Finalize methods for FV. Two stages of the FV run method are registered. The first one does the dynamics calculations, and the second adds increments from external sources that appear in the Import state. SetServices also creates a private internal state in which FV keeps invariant or auxilliary state variables, as well as pointers to the true state variables. The MAPL internal state contains the true state variables and is managed by MAPL. INTERFACE:

```
Subroutine SetServices ( gc, rc )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: gc     ! gridded component
integer, intent(out), optional     :: rc     ! return code
```

---

9 Module DynCore_GridCompMod — Dynamical Core Grid Component

USES:

```
    use ESMF                ! ESMF base class
    use MAPL_Mod            ! GEOS base class

   FV Specific Module
    use FV_StateMod, only : DynTracers       = > T_TRACERS,             &
                            DynVars          = > T_FVDYCORE_VARS,       &
                            DynGrid          = > T_FVDYCORE_GRID,       &
                            DynState         = > T_FVDYCORE_STATE,      &
                            DynInit          = > FV_InitState,          &
                            DynRun           = > FV_Run,                &
                            DynFinalize      = > FV_Finalize,           &
                            getAgridWinds    = > INTERP_DGRID_TO_AGRID, &
                            getMassFluxes    = > fv_getMassFluxes,      &
                            getOmega         = > fv_getOmega,           &
                            getPK            = > fv_getPK,              &
                            getVorticity     = > fv_getVorticity,       &
                            getEPV           = > fv_getEPV,             &
                            getPKZ           = > fv_getPKZ,             &
                            Agrid_To_Native  = > INTERP_AGRID_TO_DGRID, &
                            DYN_COLDSTART    = > COLDSTART,             &
                            DYN_DEBUG        = > DEBUG,                 &
                            HYDROSTATIC      = > FV_HYDROSTATIC,        &
                            ADIABATIC, SW_DYNAMICS, PURE_ADVECTION
    use FV_Mapz_Mod, only : ana_remap
    use m_topo_remap, only: dyn_topo_remap
```

PUBLIC MEMBER FUNCTIONS:

```
   integer :: NXQ  =  0

   public  SetServices      ! Register component methods
```

DESCRIPTION:
This module implements the Dynamical Core as an ESMF gridded component.

Overview This module contains an ESMF wrapper for a generic Dynamical Core.

Internal State FVdycore maintains an internal state consisting of the following fields: control variables

- `U`: U winds on the native grid (m/s)
- `V`: V winds on the native grid (m/s)
- `PT`: Scaled Virtual Potential Temperature (T_v/PKZ)
- `PE`: Edge pressures
- `Q`: Tracers
- `PKZ`: Consistent mean for p^κ
- `DZ`: Height thickness (Non-Hydrostatic)

as well as a GRID (to be mentioned later) and same additional run-specific variables
Note: `PT` is not updated if the flag `CONVT` is true.
The internal state is updated each time FVdycore is called.

Import State The import state consists of the tendencies of the control variables plus the surface geopotential heights:

- `DUDT`: U wind tendency on a A-grid (m/s)
- `DVDT`: V wind tendency on a A-grid (m/s)
- `DTDT`: Delta-pressure-weighted temperature tendency
- `DPEDT`: Edge pressure tendency
- `PHIS`: Surface Geopotential Heights

These are by definition on an A-grid and have an XY domain decomposition.

Export State The export state can provide the following variables:

- `U`: U winds on a A-grid (m/s)
- `V`: V winds on a A-grid (m/s)
- `U_CGRID`: U winds on a C-grid (m/s)
- `V_CGRID`: V winds on a C-grid (m/s)
- `U_DGRID`: U winds on a D-grid (m/s)
- `V_DGRID`: V winds on a D-grid (m/s)
- `T`: Temperature (K)

- Q: Tracers
- TH: Potential Temperature (K)
- ZL: Mid-Layer Heights (m)
- ZLE: Edge Heights (m)
- PLE: Edge pressures (Pa)
- PLK: P^κ at Mid-Layers
- OMEGA: Vertical pressure velocity (pa/s)
- MFX_UR: Mass-Weighted U-Wind on C-Grid (Pa m^2/s)
- MFY_UR: Mass-Weighted V-wind on C-Grid (Pa m^2/s)
- MFX: Remapped Mass-Weighted U-Wind on C-Grid (Pa m^2/s)
- MFY: Remapped Mass-Weighted V-wind on C-Grid (Pa m^2/s)
- MFZ: Remapped Vertical mass flux (kg/(m^2*s))
- MFX_A: Remapped Mass-Weighted U-Wind on A-Grid (Pa m/s)
- MFY_A: Remapped Mass-Weighted V-wind on A-Grid (Pa m/s)
- PV: Ertel's Potential Vorticity (m^2 / kg*s)
- DUDT: U-wind Tendency (m/s/s)
- DVDT: V-wind Tendency (m/s/s)
- DTDT: Mass-Weighted Temperature Tendency (Pa K/s)

All variables are on an A-grid with points at the poles, and have an XY decomposition.

Grids and Decompositions The current version supports only a 1D latitude-based decomposition of the domain (with OMP task-parallelism in the vertical, resulting in reasonable scalability on large PE configurations). In the near future it will support a 2D domain decomposition, in which import and export state are decomposed in longitude and latitude, while the internal state (for the most part) is decomposed in latitude and level. When needed, the data is redistributed ("transposed") internally.
There are two fundamental ESMF grids in use;

- GRIDXY: longitude-latitude ESMF grid (public)
- GRIDYZ: A latitude-level cross-sectional decomposition (private to this module)

PILGRIM will be used for communication until ESMF has sufficient functionality and performance to take over the task. The use of pilgrim requires a call to INIT_SPMD to set SPMD parameters, decompositions, etc.

Required Files The following files are needed for a standard restart run:

- Layout file
  - `nprxy_x, nprxy_y, npryz_y, npryz_z`: process dimensions in XY and YZ.
  - `imxy, jmxy, jmyz, kmyz`: distributions for XY and YZ
  - `iord, jord`: the order of the lon. and lat. algorithms
  - `dtime`: The large (advection) time step
  - `nsplit`: the ratio between the large and small time step (possibly zero for automatic determination),
- Restart file
  - date in standard format yy, mm, dd, hh, mm, ss
  - dimensions im, jm, km, nq
  - control variables `U, V, PT, PE, Q`
- Topography file

Future Additions

- Conservation of energy (CONSV = = .TRUE.)
- 2D decomposition (requires transposes in the coupler)
- Use r8 instead of r4 (currently supported in StopGap)

9.1 SetServices

DESCRIPTION:

SetServices registers Initialize, Run, and Finalize methods for FV. Two stages of the FV run method are registered. The first one does the dynamics calculations, and the second adds increments from external sources that appear in the Import state. SetServices also creates a private internal state in which FV keeps invariant or auxilliary state variables, as well as pointers to the true state variables. The MAPL internal state contains the true state variables and is managed by MAPL.

The component uses all three states (Import, Export and Internal), in addition to a Private (non-ESMF) Internal state. All three are managed by MAPL.

The Private Internal state contains invariant quantities defined by an FV specific routine, as well as pointers to the true state variables, kept in the MAPL Internal state. The MAPL Internal is kept at FV's real*8 precision.

The Import State conatins tendencies to be added in the second run stage, the geopotential at the lower boundary, and a bundle of Friendly tracers to be advected. The Import and Export states are both at the default precision.

INTERFACE:

```
Subroutine SetServices ( gc, rc )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: gc     ! gridded component
integer, intent(out), optional     :: rc     ! return code
```

DESCRIPTION:
Set services (register) for the FVCAM Dynamical Core Grid Component. *STATES:*

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
DUDT	IM	m s^{-2}	xyz	Center	eastward wind tendency
DVDT	IM	m s^{-2}	xyz	Center	northward wind tendency
DTDT	IM	Pa K s^{-1}	xyz	Center	delta-p weighted temperature tendency
DQVANA	IM	kg kg^{-1}	xyz	Center	specific humidity increment from analysis
DOXANA	IM	kg kg^{-1}	xyz	Center	ozone increment from analysis
DPEDT	IM	Pa s^{-1}	xyz	Edge	edge pressure tendency
PHIS	IM	m^2 s^{-2}	xy	None	surface geopotential height
TRADV	IM	unknown			advected quantities
KE	EX	J m^{-2}	xy	None	vertically integrated kinetic energy
TAVE	EX	K	xy	None	vertically averaged dry temperature
UAVE	EX	m sec^{-1}	xy	None	vertically averaged zonal wind
KEPHY	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to physics
PEPHY	EX	W m^{-2}	xy	None	total potential energy tendency due to physics
TEPHY	EX	W m^{-2}	xy	None	mountain work tendency due to physics
KEANA	EX	W m^{-2}	xy	None	total kinetic energy tendency due to analysis
PEANA	EX	W m^{-2}	xy	None	total potential energy tendency due to analysis
TEANA	EX	W m^{-2}	xy	None	mountain work tendency due to analysis
KEHOT	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to HOT

Short Name	Type	Units	Dims	Vert Loc	Long name
KEDP	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to pressure change
KEADV	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to dynamics advection
KEPG	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to pressure gradient
KEDYN	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to dynamics
PEDYN	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency due to dynamics
TEDYN	EX	W m^{-2}	xy	None	mountain work tendency due to dynamics
KECDCOR	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to cdcore
PECDCOR	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency due to cdcore
TECDCOR	EX	W m^{-2}	xy	None	mountain work tendency due to cdcore
QFIXER	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency due to CONSV
KEREMAP	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to remap
PEREMAP	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency due to remap
TEREMAP	EX	W m^{-2}	xy	None	mountain work tendency due to remap
KEGEN	EX	W m^{-2}	xy	None	vertically integrated generation of kinetic energy
DKERESIN	EX	W m^{-2}	xy	None	vertically integrated kinetic energy residual from inertial terms
DKERESPG	EX	W m^{-2}	xy	None	vertically integrated kinetic energy residual from PG terms
DMDTANA	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated mass tendency due to analysis

Short Name	Type	Units	Dims	Vert Loc	Long name
DOXDTANAINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated ozone tendency due to analysis
DQVDTANAINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated water vapor tendency due to analysis
DQLDTANAINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated liquid water tendency due to analysis
DQIDTANAINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated ice water tendency due to analysis
DMDTDYN	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated mass tendency due to dynamics
DOXDTDYNINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated ozone tendency due to dynamics
DTHVDTDYNINT	EX	K kg m^{-2} s^{-1}	xy	None	vertically integrated THV tendency due to dynamics
DTHVDTREMAP	EX	K kg m^{-2} s^{-1}	xy	None	vertically integrated THV tendency due to vertical remapping
DTHVDTCONSV	EX	K kg m^{-2} s^{-1}	xy	None	vertically integrated THV tendency due to TE conservation
DTHVDTPHYINT	EX	K kg m^{-2} s^{-1}	xy	None	vertically integrated THV tendency due to physics
DTHVDTANAINT	EX	K kg m^{-2} s^{-1}	xy	None	vertically integrated THV tendency due to analysis
DQVDTDYNINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated water vapor tendency due to dynamics
DQLDTDYNINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated liquid water tendency due to dynamics
DQIDTDYNINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated ice water tendency due to dynamics
CONVKE	EX	W m^{-2}	xy	Center	vertically integrated kinetic energy convergence
CONVTHV	EX	W m^{-2}	xy	Center	vertically integrated thetav convergence
CONVCPT	EX	W m^{-2}	xy	Center	vertically integrated enthalpy convergence
CONVPHI	EX	W m^{-2}	xy	Center	vertically integrated geopotential convergence
VORT	EX	s^{-1}	xyz	Center	vorticity
U	EX	m s^{-1}	xyz	Center	eastward wind
V	EX	m s^{-1}	xyz	Center	northward wind
T	EX	K	xyz	Center	air temperature
PL	EX	Pa	xyz	Center	mid level pressure

Short Name	Type	Units	Dims	Vert Loc	Long name
ZLE	EX	m	xyz	Edge	edge heights
ZL	EX	m	xyz	Center	mid layer heights
S	EX	m	xyz	Center	mid layer dry static energy
PLE	EX	Pa	xyz	Edge	edge pressure
TH	EX	K	xyz	Center	potential temperature
PLK	EX	Pa^{κ}	xyz	Center	mid-layer p^{κ}
W	EX	$m\ s^{-1}$	xyz	Center	vertical velocity
OMEGA	EX	$Pa\ s^{-1}$	xyz	Center	vertical pressure velocity
PTFX	EX	$K\ Pa\ m^2\ s^{-1}$	xyz	Center	pressure weighted eastward potential temperature flux unremapped
PTFY	EX	$K\ Pa\ m^2\ s^{-1}$	xyz	Center	pressure weighted northward potential temperature flux unremapped
MFX_UR	EX	$Pa\ m^2\ s^{-1}$	xyz	Center	pressure weighted eastward wind unremapped
MFY_UR	EX	$Pa\ m^2\ s^{-1}$	xyz	Center	pressure weighted northward wind unremapped
MFX	EX	$Pa\ m^2\ s^{-1}$	xyz	Center	pressure weighted eastward wind
MFY	EX	$Pa\ m^2\ s^{-1}$	xyz	Center	pressure weighted northward wind
MFZ	EX	$kg\ m^{-2}\ s^{-1}$	xyz	Edge	vertical mass flux
MFX_A	EX	$Pa\ m^2\ s^{-1}$	xyz	Center	zonal mass flux
MFY_A	EX	$Pa\ m^2\ s^{-1}$	xyz	Center	meridional mass flux
PV	EX	$m^2\ kg^{-1}\ s^{-1}$	xyz	Center	ertels isentropic potential vorticity
EPV	EX	$K\ m^2\ kg^{-1}\ s^{-1}$	xyz	Center	ertels potential vorticity
Q	EX	1	xyz	Center	specific humidity
DUDTANA	EX	m/s/s	xyz	Center	tendency of eastward wind due to analysis
DVDTANA	EX	m/s/s	xyz	Center	tendency of northward wind due to analysis
DTDTANA	EX	$K\ s^{-1}$	xyz	Center	tendency of air temperature due to analysis
DDELPDTANA	EX	$K\ s^{-1}$	xyz	Center	tendency of pressure thickness due to analysis
DUDTDYN	EX	m/s/s	xyz	Center	tendency of eastward wind due to dynamics
DVDTDYN	EX	m/s/s	xyz	Center	tendency of northward wind due to dynamics

Short Name	Type	Units	Dims	Vert Loc	Long name
DTDTDYN	EX	K s^{-1}	xyz	Center	tendency of air temperature due to dynamics
DQVDTDYN	EX	kg/kg/s	xyz	Center	tendency of specific humidity due to dynamics
DQIDTDYN	EX	kg/kg/s	xyz	Center	tendency of ice water due to dynamics
DQLDTDYN	EX	kg/kg/s	xyz	Center	tendency of liquid water due to dynamics
DOXDTDYN	EX	kg/kg/s	xyz	Center	tendency of ozone due to dynamics
PREF	EX	Pa	z	Edge	reference air pressure
AK	EX	1	z	Edge	hybrid sigma pressure a
BK	EX	1	z	Edge	hybrid sigma pressure b
PHIS	EX	m	xy	None	surface height
PS	EX	Pa	xy	None	surface pressure
TA	EX	K	xy	None	surface air temperature
QA	EX	1	xy	None	surface specific humidity
US	EX	m s^{-1}	xy	None	surface eastward wind
VS	EX	m s^{-1}	xy	None	surface northward wind
SPEED	EX	m s^{-1}	xy	None	surface wind speed
DZ	EX	m	xy	None	surface layer height
SLP	EX	Pa	xy	None	sea level pressure
H1000	EX	m	xy	None	height at 1000 mb
TROPP_EPV	EX	Pa	xy	None	tropopause pressure based on EPV estimate
TROPP_THERMAL	EX	Pa	xy	None	tropopause pressure based on thermal estimate
TROPP_BLENDED	EX	Pa	xy	None	tropopause pressure based on blended estimate
TROPT	EX	K	xy	None	tropopause temperature using blended TROPP estimate
TROPQ	EX	kg/kg	xy	None	tropopause specific humidity using blended TROPP estimate
DELP	EX	Pa	xyz	Center	pressure thickness
DELPTOP	EX	Pa	xy	None	pressure thickness at model top
U_CGRID	EX	m s^{-1}	xyz	Center	eastward wind on C-Grid
V_CGRID	EX	m s^{-1}	xyz	Center	northward wind on C-Grid
U_DGRID	EX	m s^{-1}	xyz	Center	eastward wind on native D-Grid
V_DGRID	EX	m s^{-1}	xyz	Center	northward wind on native D-Grid
TV	EX	K	xyz	Center	air virtual temperature

Short Name	Type	Units	Dims	Vert Loc	Long name
THV	EX	K/Pa$^{\kappa}$	xyz	Center	scaled virtual potential temperature
DDELPDTDYN	EX	Pa s^{-1}	xyz	Center	tendency of pressure thickness due to dynamics
UKE	EX	J m^{-1} s^{-1}	xy	None	eastward flux of atmospheric kinetic energy
VKE	EX	J m^{-1} s^{-1}	xy	None	northward flux of atmospheric kinetic energy
UCPT	EX	J m^{-1} s^{-1}	xy	None	eastward flux of atmospheric enthalpy
VCPT	EX	J m^{-1} s^{-1}	xy	None	northward flux of atmospheric enthalpy
UPHI	EX	J m^{-1} s^{-1}	xy	None	eastward flux of atmospheric potential energy
VPHI	EX	J m^{-1} s^{-1}	xy	None	northward flux of atmospheric potential energy
UQV	EX	kg m^{-1} s^{-1}	xy	None	eastward flux of atmospheric water vapor
VQV	EX	kg m^{-1} s^{-1}	xy	None	northward flux of atmospheric water vapor
UQL	EX	kg m^{-1} s^{-1}	xy	None	eastward flux of atmospheric liquid water
VQL	EX	kg m^{-1} s^{-1}	xy	None	northward flux of atmospheric liquid water
UQI	EX	kg m^{-1} s^{-1}	xy	None	eastward flux of atmospheric ice
VQI	EX	kg m^{-1} s^{-1}	xy	None	northward flux of atmospheric ice
DKE	EX	W m^{-2}	xy	None	tendency of atmosphere kinetic energy content due to dynamics
DCPT	EX	W m^{-2}	xy	None	tendency of atmosphere dry energy content due to dynamics
DPET	EX	W m^{-2}	xy	None	tendency of atmosphere topographic potential energy due to dynamics
WRKT	EX	W m^{-2}	xy	None	work done by atmosphere at top
DQV	EX	kg m^{-2} s^{-1}	xy	None	tendency of atmosphere water vapor content due to dynamics
DQL	EX	kg m^{-2} s^{-1}	xy	None	tendency of atmosphere liquid water content due to dynamics
DQI	EX	kg m^{-2} s^{-1}	xy	None	tendency of atmosphere ice content due to dynamics

Short Name	Type	Units	Dims	Vert Loc	Long name
CNV	EX	W m^{-2}	xy	None	generation of atmosphere kinetic energy content
TRIM(myTracer)	EX	1	xy	None	TRIM(myTracer)
TRIM(myTracer)	EX	1	xyz	Center	TRIM(myTracer)
VORT850	EX	m s^{-1}	xy	None	vorticity at 850 hPa
VORT700	EX	m s^{-1}	xy	None	vorticity at 700 hPa
VORT200	EX	m s^{-1}	xy	None	vorticity at 200 hPa
U850	EX	m s^{-1}	xy	None	eastward wind at 850 hPa
U700	EX	m s^{-1}	xy	None	eastward wind at 700 hPa
U500	EX	m s^{-1}	xy	None	eastward wind at 500 hPa
U250	EX	m s^{-1}	xy	None	eastward wind at 250 hPa
U200	EX	m s^{-1}	xy	None	eastward wind at 200 hPa
UTOP	EX	m s^{-1}	xy	None	eastward wind at model top
V850	EX	m s^{-1}	xy	None	northward wind at 850 hPa
V700	EX	m s^{-1}	xy	None	northward wind at 700 hPa
V500	EX	m s^{-1}	xy	None	northward wind at 500 hPa
V250	EX	m s^{-1}	xy	None	northward wind at 250 hPa
V200	EX	m s^{-1}	xy	None	northward wind at 200 hPa
VTOP	EX	m s^{-1}	xy	None	northward wind at model top
T850	EX	K	xy	None	air temperature at 850 hPa
T700	EX	K	xy	None	air temperature at 700 hPa
T500	EX	K	xy	None	air temperature at 500 hPa
T300	EX	K	xy	None	air temperature at 300 hPa
T250	EX	K	xy	None	air temperature at 250 hPa
TTOP	EX	K	xy	None	air temperature at model top
Q850	EX	kg kg^{-1}	xy	None	specific humidity at 850 hPa
Q500	EX	kg kg^{-1}	xy	None	specific humidity at 500 hPa
Q250	EX	kg kg^{-1}	xy	None	specific humidity at 250 hPa
Z700	EX	m	xy	None	geopotential height at 700 hPa
Z500	EX	m	xy	None	geopotential height at 500 hPa
Z300	EX	m	xy	None	geopotential height at 300 hPa
H850	EX	m	xy	None	height at 850 hPa
H700	EX	m	xy	None	height at 700 hPa
H500	EX	m	xy	None	height at 500 hPa
H300	EX	m	xy	None	height at 300 hPa
H250	EX	m	xy	None	height at 250 hPa
OMEGA850	EX	Pa s^{-1}	xy	None	omega at 850 hPa
OMEGA500	EX	Pa s^{-1}	xy	None	omega at 500 hPa
OMEGA200	EX	Pa s^{-1}	xy	None	omega at 200 hPa
OMEGA10	EX	Pa s^{-1}	xy	None	omega at 10 hPa
W850	EX	m s^{-1}	xy	None	w at 850 hPa
W500	EX	m s^{-1}	xy	None	w at 500 hPa
W200	EX	m s^{-1}	xy	None	w at 200 hPa

Short Name	Type	Units	Dims	Vert Loc	Long name
W10	EX	m s^{-1}	xy	None	w at 10 hPa
U50M	EX	m s^{-1}	xy	None	eastward wind at 50 meters
V50M	EX	m s^{-1}	xy	None	northward wind at 50 meters
AREA	EX	m^2	xy	None	agrid cell area
PT	EX	K $Pa^{-\kappa}$	xyz	Center	scaled potential temperature
PE	EX	Pa	xyz	Edge	air pressure
LONS	EX	radians	xy	None	Center longitudes
LATS	EX	radians	xy	None	Center latitudes
AK	IN	Pa	z	Edge	hybrid sigma pressure a
BK	IN	1	z	Edge	hybrid sigma pressure b
U	IN	m s^{-1}	xyz	Center	eastward wind
V	IN	m s^{-1}	xyz	Center	northward wind
PT	IN	K $Pa^{-\kappa}$	xyz	Center	scaled potential temperature
PE	IN	Pa	xyz	Edge	air pressure
PKZ	IN	Pa^{κ}	xyz	Center	pressure to kappa
DZ	IN	m	xyz	Center	height thickness
W	IN	m s^{-1}	xyz	Center	vertical velocity

RESOURCES:

Name	Description	Units	Default
'LAYOUT:'	name of layout file	none	'fvcore_layout.rc'

9.2 Run

DESCRIPTION:
This is the first Run stage of FV. It is the container for the dycore calculations. Subroutines from the core are invoked to do most of the work. A second run method, descibed below, adds the import tendencies from external sources to the FV variables.
In addition to computing and adding all dynamical contributions to the FV variables (i.e., winds, pressures, and temperatures), this method advects an arbitrary number of tracers. These appear in a "Friendly" bundle in the IMPORT state and are updated with the advective tendency.
INTERFACE:

```
subroutine Run(gc, import, export, clock, rc)
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: gc
type (ESMF_State),   intent(inout) :: import
type (ESMF_State),   intent(inout) :: export
type (ESMF_Clock),   intent(inout) :: clock
integer, intent(out), optional     :: rc
```

9.3 RunAddIncs

DESCRIPTION:
This is the second registered stage of FV. It calls an Fv supplied routine to add external contributions to FV's state variables. It does not touch the Friendly tracers. It also computes additional diagnostics and updates the FV internal state to reflect the added tendencies.
INTERFACE:

```
subroutine RunAddIncs(gc, import, export, clock, rc)
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: gc
type (ESMF_State),   intent(inout) :: import
type (ESMF_State),   intent(inout) :: export
type (ESMF_Clock),   intent(in)    :: clock
integer, intent(out), optional     :: rc
```

9.4 Finalize

DESCRIPTION:
Writes restarts and cleans-up through MAPL_GenericFinalize and deallocates memory from the Private Internal state. INTERFACE:

```
subroutine Finalize(gc, import, export, clock, rc)
```

ARGUMENTS:

```
type (ESMF_GridComp), intent(inout) :: gc
type (ESMF_State),    intent(inout) :: import
type (ESMF_State),    intent(inout) :: export
type (ESMF_Clock),    intent(inout) :: clock
integer, optional,    intent(  out) :: rc
```

9.5 Coldstart

DESCRIPTION:
Routine to coldstart from an isothermal state of rest. The temperature can be specified in the config, otherwise it is 300K. The surface pressure is assumed to be 1000 hPa. INTERFACE:

```
subroutine Coldstart(gc, import, export, clock, rc)

    USE jw, only : temperature, u_wind, v_wind, surface_geopotential
    USE jw, only : tracer_q, tracer_q1_q2, tracer_q3
    USE testcases_3_4_5_6, only : advection, Rossby_Haurwitz, mountain_Rossby, gravity_wave
```

ARGUMENTS:

```
    type(ESMF_GridComp), intent(inout) :: gc
    type(ESMF_State),    intent(inout) :: import
    type(ESMF_State),    intent(inout) :: export
    type (ESMF_Clock),   intent(inout) :: clock
    integer, intent(out), optional     :: rc
```

RESOURCES:

Name	Description	Units	Default
'T0:'	Value of isothermal temperature on coldstart	K	273.

10 Module FVdycore_GridCompMod — FVCAM Dynamical Core Grid Component

DESCRIPTION:
This module implements the FVCAM Dynamical Core as an ESMF gridded component.

Overview This module contains an ESMF wrapper for the Finite-Volume Dynamical Core used in the Community Atmospheric Model (FVCAM). This component will hereafter be referred to as the "FVdycore" ESMF gridded component. FVdycore consists of four subcomponents,

- `cd_core`: The C/D-grid dycore component
- `te_map`: Vertical remapping algorithm
- `trac2d`: Tracer advection

- `benergy:` Energy balance

Subsequently the ESMF component design for FV dycore will be described.

Internal State FVdycore maintains an internal state consisting of the following fields: control variables

- `U`: U winds on a D-grid (m/s)
- `V`: V winds on a D-grid (m/s)
- `PT`: Scaled Virtual Potential Temperature(T_v/PKZ)
- `PE`: Edge pressures
- `Q`: Tracers
- `PKZ`: Consistent mean for p^κ

as well as a GRID (to be mentioned later) and same additional run-specific variables (dt, iord, jord, nsplit – to be mentioned later)
Note: `PT` is not updated if the flag `CONVT` is true.
The internal state is updated each time FVdycore is called.

Import State The import state consists of the tendencies of the control variables plus the surface geopotential heights:

- `DUDT`: U wind tendency on a A-grid (m/s)
- `DVDT`: V wind tendency on a A-grid (m/s)
- `DTDT`: Delta-pressure-weighted temperature tendency
- `DPEDT`: Edge pressure tendency
- `PHIS`: Surface Geopotential Heights

These are by definition on an A-grid and have an XY domain decomposition.

Export State The export state can provide the following variables:

- `U`: U winds on a A-grid (m/s)
- `V`: V winds on a A-grid (m/s)
- `U_CGRID`: U winds on a C-grid (m/s)
- `V_CGRID`: V winds on a C-grid (m/s)
- `U_DGRID`: U winds on a D-grid (m/s)

- `V_DGRID`: V winds on a D-grid (m/s)
- `T`: Temperature (K)
- `Q`: Tracers
- `TH`: Potential Temperature (K)
- `ZL`: Mid-Layer Heights (m)
- `ZLE`: Edge Heights (m)
- `PLE`: Edge pressures (Pa)
- `PLK`: P^κ at Mid-Layers
- `OMEGA`: Vertical pressure velocity (pa/s)
- `PTFX`: Mass-Weighted PT flux on C-Grid (K Pa m^2/s)
- `PTFY`: Mass-Weighted PT flux on C-Grid (K Pa m^2/s)
- `MFX_UR`: Mass-Weighted U-Wind on C-Grid (Pa m^2/s)
- `MFY_UR`: Mass-Weighted V-wind on C-Grid (Pa m^2/s)
- `MFX`: Remapped Mass-Weighted U-Wind on C-Grid (Pa m^2/s)
- `MFY`: Remapped Mass-Weighted V-wind on C-Grid (Pa m^2/s)
- `MFZ`: Remapped Vertical mass flux (kg/(m^2*s))
- `MFX_A`: Remapped Mass-Weighted U-Wind on A-Grid (Pa m^2/s)
- `MFY_A`: Remapped Mass-Weighted V-wind on A-Grid (Pa m^2/s)
- `PV`: Ertel's Potential Vorticity (m^2 / kg*s)
- `DUDT`: U-wind Tendency (m/s/s)
- `DVDT`: V-wind Tendency (m/s/s)
- `DTDT`: Mass-Weighted Temperature Tendency (Pa K/s)
- `AREA`: Cell areas on the A-Grid (m^2, polar caps at J = 1, J = JM)

All variables are on an A-grid with points at the poles, and have an XY decomposition.

Grids and Decompositions The current version supports only a 1D latitude-based decomposition of the domain (with OMP task-parallelism in the vertical, resulting in reasonable scalability on large PE configurations). In the near future it will support a 2D domain decomposition, in which import and export state are decomposed in longitude and latitude, while the internal state (for the most part) is decomposed in latitude and level. When needed, the data is redistributed ("transposed") internally.
There are two fundamental ESMF grids in use;

- GRIDXY: longitude-latitude ESMF grid (public)
- GRIDYZ: A latitude-level cross-sectional decomposition (private to this module)

PILGRIM will be used for communication until ESMF has sufficient functionality and performance to take over the task. The use of pilgrim requires a call to `INIT_SPMD` to set SPMD parameters, decompositions, etc.
Currently, only a 1D decomposition in latitude is employed. Thus GRIDXY and GRIDYZ actually represent the same decomposition and no transposes are employed.

Required Files The following files are needed for a standard restart run:

- Layout file
  - `nprxy_x, nprxy_y, npryz_y, npryz_z`: process dimensions in XY and YZ.
  - `imxy, jmxy, jmyz, kmyz`: distributions for XY and YZ
  - `iord, jord`: the order of the lon. and lat. algorithms
  - `dtime`: The large (advection) time step
  - `nsplit`: the ratio between the large and small time step (possibly zero for automatic determination),
- Restart file
  - date in standard format yy, mm, dd, hh, mm, ss
  - dimensions im, jm, km, nq
  - control variables `U, V, PT, PE, Q`
- Topography file

Future Additions

- Conservation of energy (CONSV = = .TRUE.)
- 2D decomposition (requires transposes in the coupler)
- Use r8 instead of r4 (currently supported in StopGap)

USES:

```
use ESMF                 ! ESMF base class
use MAPL_Mod             ! GEOS base class
use G3_MPI_Util_Mod
use dynamics_vars, only : T_TRACERS, T_FVDYCORE_VARS, &
                          T_FVDYCORE_GRID, T_FVDYCORE_STATE
```

PUBLIC MEMBER FUNCTIONS:

```
public  SetServices      ! Register component methods
```

10.1 SetServices — Set services for FVCAM Dynamical Core

INTERFACE:

```
Subroutine SetServices ( gc, rc )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: gc     ! gridded component
integer, intent(out), optional     :: rc     ! return code
```

DESCRIPTION:
Set services (register) for the FVCAM Dynamical Core Grid Component. *STATES:*

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
DUDT	IM	m s^{-2}	xyz	Center	eastward wind tendency
DVDT	IM	m s^{-2}	xyz	Center	northward wind tendency
DTDT	IM	Pa K s^{-1}	xyz	Center	delta-p weighted temperature tendency
DQVANA	IM	kg kg^{-1}	xyz	Center	specific humidity increment from analysis
DOXANA	IM	kg kg^{-1}	xyz	Center	ozone increment from analysis
DPEDT	IM	Pa s^{-1}	xyz	Edge	edge pressure tendency
PHIS	IM	m^2 s^{-2}	xy	None	surface geopotential height
TRADV	IM	unknown			advected quantities
KE	EX	J m^{-2}	xy	None	vertically integrated kinetic energy

Short Name	Type	Units	Dims	Vert Loc	Long name
TAVE	EX	K	xy	None	vertically averaged dry temperature
UAVE	EX	m s^{-1}	xy	None	vertically averaged zonal wind
KEPHY	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to physics
PEPHY	EX	W m^{-2}	xy	None	total potential energy tendency due to physics
TEPHY	EX	W m^{-2}	xy	None	mountain work tendency due to physics
KEANA	EX	W m^{-2}	xy	None	total kinetic energy tendency due to analysis
PEANA	EX	W m^{-2}	xy	None	total potential energy tendency due to analysis
TEANA	EX	W m^{-2}	xy	None	mountain work tendency due to analysis
KEHOT	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to HOT
KEDP	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to pressure change
KEADV	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to dynamics advection
KEPG	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to pressure gradient
KEDYN	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to dynamics
PEDYN	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency due to dynamics
TEDYN	EX	W m^{-2}	xy	None	mountain work tendency due to dynamics
KECDCOR	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to cdcore
PECDCOR	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency due to cdcore
TECDCOR	EX	W m^{-2}	xy	None	mountain work tendency due to cdcore

Short Name	Type	Units	Dims	Vert Loc	Long name
QFIXER	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency due to CONSV
KEREMAP	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency due to remap
PEREMAP	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency due to remap
TEREMAP	EX	W m^{-2}	xy	None	mountain work tendency due to remap
KEGEN	EX	W m^{-2}	xy	None	vertically integrated generation of kinetic energy
DKERESIN	EX	W m^{-2}	xy	None	vertically integrated kinetic energy residual from inertial terms
DKERESPG	EX	W m^{-2}	xy	None	vertically integrated kinetic energy residual from PG terms
DMDTANA	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated mass tendency due to analysis
DOXDTANAINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated ozone tendency due to analysis
DQVDTANAINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated water vapor tendency due to analysis
DQLDTANAINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated liquid water tendency due to analysis
DQIDTANAINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated ice water tendency due to analysis
DMDTDYN	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated mass tendency due to dynamics
DOXDTDYNINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated ozone tendency due to dynamics
DTHVDTDYNINT	EX	K kg m^{-2} s^{-1}	xy	None	vertically integrated THV tendency due to dynamics
DTHVDTREMAP	EX	K kg m^{-2} s^{-1}	xy	None	vertically integrated THV tendency due to vertical remapping
DTHVDTCONSV	EX	K kg m^{-2} s^{-1}	xy	None	vertically integrated THV tendency due to TE conservation
DTHVDTPHYINT	EX	K kg m^{-2} s^{-1}	xy	None	vertically integrated THV tendency due to physics
DTHVDTANAINT	EX	K kg m^{-2} s^{-1}	xy	None	vertically integrated THV tendency due to analysis

Short Name	Type	Units	Dims	Vert Loc	Long name
DQVDTDYNINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated water vapor tendency due to dynamics
DQLDTDYNINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated liquid water tendency due to dynamics
DQIDTDYNINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated ice water tendency due to dynamics
CONVKE	EX	W m^{-2}	xy	Center	vertically integrated kinetic energy convergence
CONVTHV	EX	W m^{-2}	xy	Center	vertically integrated thetav convergence
CONVCPT	EX	W m^{-2}	xy	Center	vertically integrated enthalpy convergence
CONVPHI	EX	W m^{-2}	xy	Center	vertically integrated geopotential convergence
U	EX	m s^{-1}	xyz	Center	eastward wind
V	EX	m s^{-1}	xyz	Center	northward wind
T	EX	K	xyz	Center	air temperature
PL	EX	Pa	xyz	Center	mid level pressure
ZLE	EX	m	xyz	Edge	edge heights
ZL	EX	m	xyz	Center	mid layer heights
S	EX	m	xyz	Center	mid layer dry static energy
PLE	EX	Pa	xyz	Edge	edge pressure
TH	EX	K	xyz	Center	potential temperature
PLK	EX	Pa^{κ}	xyz	Center	mid layer p^{κ}
OMEGA	EX	Pa s^{-1}	xyz	Center	vertical pressure velocity
PTFX	EX	K Pa m^2 s^{-1}	xyz	Center	pressure weighted eastward potential temperature flux unremapped
PTFY	EX	K Pa m^2 s^{-1}	xyz	Center	pressure weighted northward potential temperature flux unremapped
MFX_UR	EX	Pa m^2 s^{-1}	xyz	Center	pressure weighted eastward wind unremapped
MFY_UR	EX	Pa m^2 s^{-1}	xyz	Center	pressure weighted northward wind unremapped
MFX	EX	Pa m^2 s^{-1}	xyz	Center	pressure weighted eastward wind
MFY	EX	Pa m^2 s^{-1}	xyz	Center	pressure weighted northward wind
MFZ	EX	kg m^{-2} s^{-1}	xyz	Edge	vertical mass flux
MFX_A	EX	Pa m^2 s^{-1}	xyz	Center	zonal mass flux
MFY_A	EX	Pa m^2 s^{-1}	xyz	Center	meridional mass flux

Short Name	Type	Units	Dims	Vert Loc	Long name
PV	EX	$m^2\ kg^{-1}\ s^{-1}$	xyz	Center	ertels isentropic potential vorticity
EPV	EX	$K\ m^2\ kg^{-1}\ s^{-1}$	xyz	Center	ertels potential vorticity
Q	EX	$kg\ kg^{-1}$	xyz	Center	specific humidity
DUDTANA	EX	$m\ s^{-2}$	xyz	Center	tendency of eastward wind due to analysis
DVDTANA	EX	$m\ s^{-2}$	xyz	Center	tendency of northward wind due to analysis
DTDTANA	EX	$K\ s^{-1}$	xyz	Center	tendency of air temperature due to analysis
DDELPDTANA	EX	$K\ s^{-1}$	xyz	Center	tendency of pressure thickness due to analysis
DUDTDYN	EX	$m\ s^{-2}$	xyz	Center	tendency of eastward wind due to dynamics
DVDTDYN	EX	$m\ s^{-2}$	xyz	Center	tendency of northward wind due to dynamics
DTDTDYN	EX	$K\ s^{-1}$	xyz	Center	tendency of air temperature due to dynamics
DQVDTDYN	EX	$kg\ kg^{-1}\ s^{-1}$	xyz	Center	tendency of specific humidity due to dynamics
DQIDTDYN	EX	$kg\ kg^{-1}\ s^{-1}$	xyz	Center	tendency of ice water due to dynamics
DQLDTDYN	EX	$kg\ kg^{-1}\ s^{-1}$	xyz	Center	tendency of liquid water due to dynamics
DOXDTDYN	EX	$kg\ kg^{-1}\ s^{-1}$	xyz	Center	tendency of ozone due to dynamics
PREF	EX	Pa	z	Edge	reference air pressure
AK	EX	1	z	Edge	hybrid sigma pressure a
BK	EX	1	z	Edge	hybrid sigma pressure b
PS	EX	Pa	xy	None	surface pressure
TA	EX	K	xy	None	surface air temperature
QA	EX	$kg\ kg^{-1}$	xy	None	surface specific humidity
US	EX	$m\ s^{-1}$	xy	None	surface eastward wind
VS	EX	$m\ s^{-1}$	xy	None	surface northward wind
SPEED	EX	$m\ s^{-1}$	xy	None	surface wind speed
DZ	EX	m	xy	None	surface layer height
SLP	EX	Pa	xy	None	sea level pressure
H1000	EX	m	xy	None	height at 1000 mb
TROPP_EPV	EX	Pa	xy	None	tropopause pressure based on EPV estimate
TROPP_THERMAL	EX	Pa	xy	None	tropopause pressure based on thermal estimate

Short Name	Type	Units	Dims	Vert Loc	Long name
TROPP_BLENDED	EX	Pa	xy	None	tropopause pressure based on blended estimate
TROPT	EX	K	xy	None	tropopause temperature using blended TROPP estimate
TROPQ	EX	kg kg^{-1}	xy	None	tropopause specific humidity using blended TROPP estimate
DELP	EX	Pa	xyz	Center	pressure thickness
U_CGRID	EX	m s^{-1}	xyz	Center	eastward wind on C-Grid
V_CGRID	EX	m s^{-1}	xyz	Center	northward wind on C-Grid
U_DGRID	EX	m s^{-1}	xyz	Center	eastward wind on native D-Grid
V_DGRID	EX	m s^{-1}	xyz	Center	northward wind on native D-Grid
TV	EX	K	xyz	Center	air virtual temperature
THV	EX	K/Pa^{κ}	xyz	Center	scaled virtual potential temperature
DDELPDTDYN	EX	Pa s^{-1}	xyz	Center	tendency of pressure thickness due to dynamics
UKE	EX	J m^{-1} s^{-1}	xy	None	eastward flux of atmospheric kinetic energy
VKE	EX	J m^{-1} s^{-1}	xy	None	northward flux of atmospheric kinetic energy
UCPT	EX	J m^{-1} s^{-1}	xy	None	eastward flux of atmospheric enthalpy
VCPT	EX	J m^{-1} s^{-1}	xy	None	northward flux of atmospheric enthalpy
UPHI	EX	J m^{-1} s^{-1}	xy	None	eastward flux of atmospheric potential energy
VPHI	EX	J m^{-1} s^{-1}	xy	None	northward flux of atmospheric potential energy
UQV	EX	kg m^{-1} s^{-1}	xy	None	eastward flux of atmospheric water vapor
VQV	EX	kg m^{-1} s^{-1}	xy	None	northward flux of atmospheric water vapor
UQL	EX	kg m^{-1} s^{-1}	xy	None	eastward flux of atmospheric liquid water
VQL	EX	kg m^{-1} s^{-1}	xy	None	northward flux of atmospheric liquid water
UQI	EX	kg m^{-1} s^{-1}	xy	None	eastward flux of atmospheric ice
VQI	EX	kg m^{-1} s^{-1}	xy	None	northward flux of atmospheric ice
DKE	EX	W m^{-2}	xy	None	tendency of atmosphere kinetic energy content due to dynamics

Short Name	Type	Units	Dims	Vert Loc	Long name
DCPT	EX	W m^{-2}	xy	None	tendency of atmosphere dry energy content due to dynamics
DPET	EX	W m^{-2}	xy	None	tendency of atmosphere topographic potential energy due to dynamics
WRKT	EX	W m^{-2}	xy	None	work done by atmosphere at top
DQV	EX	kg m^{-2} s^{-1}	xy	None	tendency of atmosphere water vapor content due to dynamics
DQL	EX	kg m^{-2} s^{-1}	xy	None	tendency of atmosphere liquid water content due to dynamics
DQI	EX	kg m^{-2} s^{-1}	xy	None	tendency of atmosphere ice content due to dynamics
CNV	EX	W m^{-2}	xy	None	generation of atmosphere kinetic energy content
U850	EX	m s^{-1}	xy	None	eastward wind at 850 hPa
U500	EX	m s^{-1}	xy	None	eastward wind at 500 hPa
U250	EX	m s^{-1}	xy	None	eastward wind at 250 hPa
V850	EX	m s^{-1}	xy	None	northward wind at 850 hPa
V500	EX	m s^{-1}	xy	None	northward wind at 500 hPa
V250	EX	m s^{-1}	xy	None	northward wind at 250 hPa
T850	EX	K	xy	None	air temperature at 850 hPa
T500	EX	K	xy	None	air temperature at 500 hPa
T250	EX	K	xy	None	air temperature at 250 hPa
Q850	EX	kg kg^{-1}	xy	None	specific humidity at 850 hPa
Q500	EX	kg kg^{-1}	xy	None	specific humidity at 500 hPa
Q250	EX	kg kg^{-1}	xy	None	specific humidity at 250 hPa
H850	EX	m	xy	None	height at 850 hPa
H500	EX	m	xy	None	height at 500 hPa
H250	EX	m	xy	None	height at 250 hPa
OMEGA500	EX	Pa s^{-1}	xy	None	omega at 500 hPa
U50M	EX	m s^{-1}	xy	None	eastward wind at 50 meters
V50M	EX	m s^{-1}	xy	None	northward wind at 50 meters
AREA	EX	m^2	xy	None	agrid cell area
PT	EX	K $Pa^{-\kappa}$	xyz	Center	scaled potential temperature
PE	EX	Pa	xyz	Edge	air pressure
AK	IN	Pa	z	Edge	hybrid sigma pressure a
BK	IN	1	z	Edge	hybrid sigma pressure b
U	IN	m s^{-1}	xyz	Center	eastward wind
V	IN	m s^{-1}	xyz	Center	northward wind
PT	IN	K $Pa^{-\kappa}$	xyz	Center	scaled potential temperature
PE	IN	Pa	xyz	Edge	air pressure
PKZ	IN	Pa^{κ}	xyz	Center	pressure to kappa

10.2 Finalize

DESCRIPTION:
Writes restarts and cleans-up through MAPL_GenericFinalize and deallocates memory from the Private Internal state. INTERFACE:

```
subroutine Finalize(gc, import, export, clock, rc)
```

USES:

```
use dynamics_vars, only : dynamics_clean
```

ARGUMENTS:

```
type (ESMF_GridComp), intent(inout) :: gc
type (ESMF_State),    intent(inout) :: import
type (ESMF_State),    intent(inout) :: export
type (ESMF_Clock),    intent(inout) :: clock
integer, optional,    intent(  out) :: rc
```

10.3 Coldstart

DESCRIPTION:
Routine to coldstart from an isothermal state of rest. The temperature can be specified in the config, otherwise it is 300K. The surface pressure is assumed to be 1000 hPa. INTERFACE:

```
subroutine Coldstart(gc, import, export, clock, rc)
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: gc
type(ESMF_State),    intent(inout) :: import
type(ESMF_State),    intent(inout) :: export
type (ESMF_Clock),   intent(in)    :: clock
integer, intent(out), optional     :: rc
```

RESOURCES:

Name	Description	Units	Default
'T0:'	Value of isothermal temperature on coldstart	K	300.

11 Module GAAS_GridCompMod - Implements GEOS-5 Aerosol Assimation

INTERFACE:

```
MODULE GAASGridCompMod
```

USES:

```
Use ESMF
Use MAPL_Mod
Use MAPL_MaxMinMod
Use m_StrTemplate

Use  LDE_Mod
Use  Chem_SimpleBundleMod
Use  Chem_RegistryMod
Use  Chem_MieMod
Use  Chem_AodMod

IMPLICIT NONE
PRIVATE
```

PUBLIC MEMBER FUNCTIONS:

```
PUBLIC SetServices
```

DESCRIPTION:
`GAAS_GridComp` is an ESMF gridded component implementing the GEOS-5 Aerosol Analysis System (GAAS).
Developed for GEOS-5 release Fortuna 2.3 and later. REVISION HISTORY:

```
30Nov2010 da Silva  Initial version.
```

11.1 SetServices — Sets IRF services for the GAAS Grid Component

INTERFACE:

```
SUBROUTINE SetServices ( GC, RC )
```

ARGUMENTS:

```
 type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
 integer, optional                  :: RC  ! return code
```

DESCRIPTION:
Sets Initialize, Run and Finalize services. REVISION HISTORY:

```
30Nov2010 da Silva  Initial version.
```

11.2 Initialize_ — Initialize GAAS

INTERFACE:

```
SUBROUTINE Initialize_ ( GC, IMPORT, EXPORT, CLOCK, rc )
```

USES:

INPUT PARAMETERS:

```
type(ESMF_Clock),  intent(inout) :: CLOCK     ! The clock
```

OUTPUT PARAMETERS:

```
type(ESMF_GridComp), intent(inout)  :: GC     ! Grid Component
type(ESMF_State), intent(inout) :: IMPORT     ! Import State
type(ESMF_State), intent(inout) :: EXPORT     ! Export State
integer, intent(out)            ::  rc        ! Error return code:
                                              !  0 - all is well
                                              !  1 -
```

DESCRIPTION:
This is a simple ESMF wrapper. REVISION HISTORY:

```
30Nov2010 da Silva  Initial version.
```

11.3 Run_ — Runs GAAS

INTERFACE:

```
SUBROUTINE Run_ ( gc, IMPORT, EXPORT, CLOCK, rc )
```

USES:

INPUT PARAMETERS:

```
type(ESMF_Clock),  intent(inout) :: CLOCK     ! The clock
```

OUTPUT PARAMETERS:

```
type(ESMF_GridComp), intent(inout)  :: GC     ! Grid Component
type(ESMF_State), intent(inout) :: IMPORT     ! Import State
type(ESMF_State), intent(inout) :: EXPORT     ! Export State
integer, intent(out) ::  rc                   ! Error return code:
                                              !  0 - all is well
                                              !  1 -
```

DESCRIPTION:
This is a simple ESMF wrapper. REVISION HISTORY:

```
30Nov2010 da Silva  Initial version.
```

11.4 Finalize_ — Finalize GAAS

INTERFACE:

```
SUBROUTINE Finalize_ ( GC, IMPORT, EXPORT, CLOCK, rc )
```

USES:

INPUT PARAMETERS:

```
type(ESMF_Clock),  intent(inout) :: CLOCK      ! The clock
```

OUTPUT PARAMETERS:

```
type(ESMF_GridComp), intent(inout)  :: gc     ! Grid Component
type(ESMF_State), intent(inout) :: IMPORT     ! Import State
type(ESMF_State), intent(inout) :: EXPORT     ! Export State
integer, intent(out) ::  rc                   ! Error return code:
                                              !  0 - all is well
                                              !  1 -
```

DESCRIPTION:
This is a simple ESMF wrapper. REVISION HISTORY:

```
30Nov2010 da Silva  Initial version.
```

12 Module GEOS_AgcmGridCompMod – A Module to combine Supedynamics and Physics Gridded Components

INTERFACE:

```
module GEOSAgcmGridCompMod
```

USES:

```
use ESMF
use MAPL_Mod
use GEOS_TopoGetMod

use GEOS_superdynGridCompMod,  only:  SDYN_SetServices  = > SetServices
use GEOS_physicsGridCompMod,   only:  PHYS_SetServices  = > SetServices
use MAPL_OrbGridCompMod,       only:  ORB_SetServices  = > SetServices
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices

= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
```

DESCRIPTION:
This gridded component (GC) combines the Superdynamics GC, and Physics GC into a new composite Agcm GC.

```
DUDT .... Mass-Weighted U-Wind      Tendency (Pa m /s)
DVDT .... Mass-Weighted V-Wind      Tendency (Pa m /s)
DPEDT ... Edge-Pressure             Tendency (Pa   /s)
DTDT .... Mass-Weighted Temperature Tendency (Pa K /s)
TRACER .. Friendly Tracers                   (unknown)
```

12.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer, optional                  :: RC  ! return code
```

DESCRIPTION:
The SetServices for the Physics GC needs to register its Initialize and Run. It uses the MAPL_Generic construct for defining state specs and couplings among its children. In addition, it creates the children GCs (SURF, CHEM, RADIATION, MOIST, TURBULENCE) and runs their respective SetServices. *STATES:*

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
DUDT	IM	m s^{-1}	xyz	Center	eastward wind analysis increment
DVDT	IM	m s^{-1}	xyz	Center	northward wind analysis increment
DTDT	IM	K	xyz	Center	temperature analysis increment
DPEDT	IM	Pa	xyz	Edge	edge pressure analysis increment
DQVDT	IM	kg kg^{-1}	xyz	Center	specific humidity analysis increment
DO3DT	IM	mol mol^{-1}	xyz	Center	ozone analysis increment
DTSDT	IM	K	xy	None	skin temperature increment
DUDT	IN	m s^{-2}	xyz	Center	eastward wind bias tendency
DVDT	IN	m s^{-2}	xyz	Center	northward wind bias tendency
DTDT	IN	K s^{-1}	xyz	Center	temperature bias tendency
DPEDT	IN	Pa s^{-1}	xyz	Edge	edge pressure bias tendency
DQVDT	IN	kg kg^{-1} s^{-1}	xyz	Center	specific humidity bias tendency
DO3DT	IN	mol mol^{-1} s^{-1}	xyz	Center	ozone bias tendency
DTSDT	IN	K s^{-1}	xy	None	skin temperature tendency
DUDT_ANA	EX	m s^{-2}	xyz	Center	total eastward wind analysis tendency
DVDT_ANA	EX	m s^{-2}	xyz	Center	total northward wind analysis tendency
DTDT_ANA	EX	K s^{-1}	xyz	Center	total temperature analysis tendency
DPEDT_ANA	EX	Pa s^{-1}	xyz	Edge	total edge pressure analysis tendency
DQVDT_ANA	EX	kg kg^{-1} s^{-1}	xyz	Center	total specific humidity analysis tendency

Short Name	Type	Units	Dims	Vert Loc	Long name
DO3DT_ANA	EX	mol mol^{-1} s^{-1}	xyz	Center	total ozone analysis tendency
DTSDT_ANA	EX	K s^{-1}	xy	None	total skin temperature tendency
DTHVDTFILINT	EX	K kg m^{-2} s^{-1}	xy	None	vertically integrated thv adjustment from filling
PERES	EX	W m^{-2}	xy	None	vertically integrated cpt tendency residual
PEFILL	EX	W m^{-2}	xy	None	vertically integrated cpt adjustment from filling
QTFILL	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated total water adjustment from filling
QVFILL	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated qv adjustment from filling
QLFILL	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated ql adjustment from filling
QIFILL	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated qi adjustment from filling
OXFILL	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated ox adjustment from filling
TROPP_EPV	EX	Pa	xy	None	tropopause pressure based on EPV estimate
TROPP_THERMAL	EX	Pa	xy	None	tropopause pressure based on thermal estimate
TROPP_BLENDED	EX	Pa	xy	None	tropopause pressure based on blended estimate
TROPT	EX	K	xy	None	tropopause temperature using blended TROPP estimate
TROPQ	EX	kg kg^{-1}	xy	None	tropopause specific humidity using blended TROPP estimate
TQV	EX	kg m^{-2}	xy	None	total precipitable water vapor
TQI	EX	kg m^{-2}	xy	None	total precipitable ice water
TQL	EX	kg m^{-2}	xy	None	total precipitable liquid water
TOX	EX	kg m^{-2}	xy	None	total column odd oxygen
MASS	EX	kg m^{-2}	xy	None	atmospheric mass
KE	EX	J m^{-2}	xy	None	vertically integrated kinetic energy
CPT	EX	J m^{-2}	xy	None	vertically integrated enthalpy
THV	EX	K	xy	None	vertically integrated virtual potential temperature
QLTOT	EX	kg kg^{-1}	xyz	Center	mass fraction of cloud liquid water

Short Name	Type	Units	Dims	Vert Loc	Long name
QITOT	EX	kg kg^{-1}	xyz	Center	mass fraction of cloud ice water
PHIS	EX	m^2 s^{-2}	xy	None	surface
SGH	EX	m	xy	None	isotropic
GWDVARX	EX	m^2	xy	None	east-west
GWDVARY	EX	m^2	xy	None	north-south
GWDVARXY	EX	m^2	xy	None	SW-NE
GWDVARYX	EX	m^2	xy	None	NW-SE
TRBVAR	EX	m^2	xy	None	isotropic
VARFLT	EX	m^2	xy	None	isotropic
AREA	EX				
AK	EX				
BK	EX				
PLE	EX				
PS	EX				
PE	EX				
PT	EX				
TV	EX				
T	EX				
U	EX				
V	EX				
U_DGRID	EX				
V_DGRID	EX				
O3PPMV	EX				
OX	EX				
Q	EX				
QCTOT	EX				
U10N	EX				
V10N	EX				
SNOMAS	EX				
WET1	EX				
TSOIL1	EX				
LWI	EX				
Z0	EX				
TS	EX				
TRANA	EX				
FRLAND	EX				
FRLANDICE	EX				
FRLAKE	EX				
FROCEAN	EX				
FRACI	EX				

13 Module GEOS_Catch — ESMF gridded component implementing Catchment LSM

DESCRIPTION:

`Catch` is a gridded component to compute the energy and water fluxes due to land-surface processes, using the Catchment LSM of Koster et al. (2000). All of its calculations are done in a tile space defined by the inherited location stream. It has a two-stage run method. The first stage obtains drag coefficients at all the subtiles and defines effective tile-mean surface quantities. The second stage calls the Catchment LSM. `Catch` has no children.

USES:

```
use sfclayer  ! using module that contains sfc layer code
use ESMF
use GEOS_Mod
use DragCoefficientsMod
use CATCHMENT_MODEL
USE STIEGLITZSNOW
USE CATCH_CONSTANTS, ONLY : SNWALB_VISMAX, SNWALB_NIRMAX, SLOPE, N_snow, N_constit,RHOFS
USE MAPL_BaseMod
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

13.1 SetServices – Sets ESMF services for component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp),intent(INOUT) :: GC
integer, optional,  intent(  OUT) :: RC
```

DESCRIPTION:

This version uses GEOS_GenericSetServices, overriding only the run method. It also relies on MAPL_Generic to handle data services. *STATES:*

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
PS	IM	Pa	tile	None	surface pressure
TA	IM	K	tile	None	surface air temperature
QA	IM	kg kg^{-1}	tile	None	surface air specific humidity
UU	IM	m s^{-1}	tile	None	surface wind speed
UWINDLMTILE	IM	m s^{-1}	tile	None	levellm uwind
VWINDLMTILE	IM	m s^{-1}	tile	None	levellm vwind
PCU	IM	kg m^{-2} s^{-1}	tile	None	liquid water convective precipitation
PLS	IM	kg m^{-2} s^{-1}	tile	None	liquid water large scale precipitation
SNO	IM	kg m^{-2} s^{-1}	tile	None	snowfall
DRPAR	IM	W m^{-2}	tile	None	surface downwelling par beam flux
DFPAR	IM	W m^{-2}	tile	None	surface downwelling par diffuse flux
DRNIR	IM	W m^{-2}	tile	None	surface downwelling nir beam flux
DFNIR	IM	W m^{-2}	tile	None	surface downwelling nir diffuse flux
DRUVR	IM	W m^{-2}	tile	None	surface downwelling uvr beam flux
DFUVR	IM	W m^{-2}	tile	None	surface downwelling uvr diffuse flux
LWDNSRF	IM	W m^{-2}	tile	None	surface downwelling longwave flux
ALW	IM	W m^{-2}	tile	None	linearization of surface upwelling longwave flux
BLW	IM	W_m^{-2} K^{-1}	tile	None	linearization of surface upwelling longwave flux
LAI	IM	1	tile	None	leaf area index
GRN	IM	1	tile	None	greeness fraction
EVAP	IM	kg m^{-2} s^{-1}	tile	None	evaporation
DEVAP	IM	kg m^{-2} s^{-1}	tile	None	derivative of evaporation wrt QS
SH	IM	W m^{-2}	tile	None	upward sensible heat flux
DSH	IM	W m^{-2} K^{-1}	tile	None	derivative of sensible heat wrt Ts
DZ	IM	m	tile	None	surface layer height
ROOTL	IM	m	tile	None	vegetation root length
Z2CH	IM	m	tile	None	canopy height
THATM	IM	K	tile	None	effective surface skin temperature

Short Name	Type	Units	Dims	Vert Loc	Long name
QHATM	IM	kg kg^{-1}	tile	None	effective surface specific humidity
CTATM	IM	kg m^{-2} s^{-1}	tile	None	surface exchange coefficient for heat
CQATM	IM	kg m^{-2} s^{-1}	tile	None	surface exchange coefficient for moisture
ITY	IM	1	tile	None	vegetation type
BF1	IN	kg m^{-4}	tile	None	topo baseflow param 1
BF2	IN	m	tile	None	topo baseflow param 2
BF3	IN	log(m)	tile	None	topo baseflow param 3
VGWMAX	IN	kg m^{-2}	tile	None	max rootzone water content
CDCR1	IN	kg m^{-2}	tile	None	moisture threshold
CDCR2	IN	kg m^{-2}	tile	None	max water content
PSIS	IN	m	tile	None	saturated matric potential
BEE	IN	1	tile	None	clapp hornberger b
POROS	IN	1	tile	None	soil porosity
WPWET	IN	1	tile	None	wetness at wilting point
COND	IN	m s^{-1}	tile	None	sfc sat hydraulic conduct
GNU	IN	m^{-1}	tile	None	vertical transmissivity
ARS1	IN	m^{2} kg^{-1}	tile	None	wetness param 1
ARS2	IN	m^{2} kg^{-1}	tile	None	wetness param 2
ARS3	IN	m^{4} kg^{-2}	tile	None	wetness param 3
ARA1	IN	m^{2} kg^{-1}	tile	None	shape param 1
ARA2	IN	1	tile	None	shape param 2
ARA3	IN	m^{2} kg^{-1}	tile	None	shape param 3
ARA4	IN	1	tile	None	shape param 4
ARW1	IN	m^{2} kg^{-1}	tile	None	min theta param 1
ARW2	IN	m^{2} kg^{-1}	tile	None	min theta param 2
ARW3	IN	m^{4} kg^{-2}	tile	None	min theta param 3
ARW4	IN	1	tile	None	min theta param 4
TSA1	IN	1	tile	None	water transfer param 1
TSA2	IN	1	tile	None	water transfer param 2
TSB1	IN	1	tile	None	water transfer param 3
TSB2	IN	1	tile	None	water transfer param 4
ATAU	IN	1	tile	None	water transfer param 5
BTAU	IN	1	tile	None	water transfer param 6
OLD_ITY	IN	1	tile	None	Placeholder.
TC	IN	K	TileTile	None	canopy temperature
QC	IN	kg kg^{-1}	TileTile	None	canopy specific humidity
CAPAC	IN	kg m^{-2}	tile	None	interception reservoir capac
CATDEF	IN	kg m^{-2}	tile	None	catchment deficit
RZEXC	IN	kg m^{-2}	tile	None	root zone excess
SRFEXC	IN	kg m^{-2}	tile	None	surface excess

Short Name	Type	Units	Dims	Vert Loc	Long name
GHTCNT1	IN	J m^{-2}	tile	None	soil heat content layer 1
GHTCNT2	IN	J_m^{-2}	tile	None	soil heat content layer 2
GHTCNT3	IN	J m^{-2}	tile	None	soil heat content layer 3
GHTCNT4	IN	J m^{-2}	tile	None	soil heat content layer 4
GHTCNT5	IN	J m^{-2}	tile	None	soil heat content layer 5
GHTCNT6	IN	J m^{-2}	tile	None	soil heat content layer 6
TSURF	IN	K	tile	None	mean catchment temp incl snw
WESNN1	IN	kg m^{-2}	tile	None	snow mass layer 1
WESNN2	IN	kg m^{-2}	tile	None	snow mass layer 2
WESNN3	IN	kg m^{-2}	tile	None	snow mass layer 3
HTSNNN1	IN	J m^{-2}	tile	None	heat content snow layer 1
HTSNNN2	IN	J m^{-2}	tile	None	heat content snow layer 2
HTSNNN3	IN	J m^{-2}	tile	None	heat content snow layer 3
SNDZN1	IN	m	tile	None	snow depth layer 1
SNDZN2	IN	m	tile	None	snow depth layer 2
SNDZN3	IN	m	tile	None	snow depth layer 3
CH	IN	kg m^{-2} s^{-1}	TileTile	None	surface heat exchange coefficient
CM	IN	kg m^{-2} s^{-1}	TileTile	None	surface momentum exchange coefficient
CQ	IN	kg m^{-2} s^{-1}	TileTile	None	surface moisture exchange coffiecient
FR	IN	1	TileTile	None	subtile fractions
WW	IN	m^{2} s^{-2}	TileTile	None	vertical velocity scale squared
EVAPOUT	EX	kg m^{-2} s^{-1}	tile	None	evaporation
SUBLIM	EX	kg m^{-2} s^{-1}	tile	None	sublimation
SHOUT	EX	W m^{-2}	tile	None	upward sensible heat flux
RUNOFF	EX	kg m^{-2} s^{-1}	tile	None	runoff flux
EVPINT	EX	W m^{-2}	tile	None	interception loss energy flux
EVPSOI	EX	W m^{-2}	tile	None	baresoil evap energy flux
EVPVEG	EX	W m^{-2}	tile	None	transpiration energy flux
EVPICE	EX	W m^{-2}	tile	None	snow ice evaporation energy flux
WAT10CM	EX	kg m^{-2}	tile	None	soil
WATSOI	EX	kg m^{-2}	tile	None	totoal
ICESOI	EX	kg m^{-2}	tile	None	soil
EVPSNO	EX	W m^{-2}	tile	None	snowpack evaporation energy flux
BASEFLOW	EX	kg m^{-2} s^{-1}	tile	None	baseflow flux
RUNSURF	EX	kg m^{-2} s^{-1}	tile	None	surface runoff flux
SMELT	EX	kg m^{-2} s^{-1}	tile	None	snowmelt flux
HLWUP	EX	W m^{-2}	tile	None	surface outgoing longwave flux

Short Name	Type	Units	Dims	Vert Loc	Long name
LWNDSRF	EX	W m^{-2}	tile	None	surface net downward longwave flux
SWNDSRF	EX	W m^{-2}	tile	None	surface net downward shortwave flux
HLATN	EX	W m^{-2}	tile	None	total latent energy flux
QINFIL	EX	kg m^{-2} s^{-1}	tile	None	rainwater infiltration flux
AR1	EX	1	tile	None	areal fraction saturated zone
AR2	EX	1	tile	None	areal fraction transpiration zone
RZEQ	EX	kg m^{-2}	tile	None	root zone equilibrium moisture
GHFLX	EX	W m^{-2}	tile	None	ground energy flux
TPSURF	EX	K	tile	None	ave catchment temp incl snw
TPSNOW	EX	K	tile	None	temperature top snow layer
TPUNST	EX	K	tile	None	temperature unsaturated zone
TPSAT	EX	K	tile	None	temperature saturated zone
TPWLT	EX	K	tile	None	temperature wilted zone
ASNOW	EX	1	tile	None	fractional area of land snowcover
SHSNOW	EX	W m^{-2}	tile	None	downward heat flux into snow
AVETSNOW	EX	K	tile	None	averaged snow temperature
FRSAT	EX	1	tile	None	fractional area of saturated zone
FRUST	EX	1	tile	None	fractional area of unsaturated zone
FRWLT	EX	1	tile	None	fractional area of wilting zone
SNOWMASS	EX	kg m^{-2}	tile	None	snow mass
SNOWDP	EX	m	tile	None	snow depth
WET1	EX	1	tile	None	surface soil wetness
WET2	EX	1	tile	None	root zone soil wetness
WET3	EX	1	tile	None	ave prof soil moisture
WCSF	EX	m3 m^{-3}	tile	None	water surface layer
WCRZ	EX	m3 m^{-3}	tile	None	water root zone
WCPR	EX	m3 m^{-3}	tile	None	water ave prof
TP1	EX	C	tile	None	soil temperatures layer 1
TP2	EX	C	tile	None	soil temperatures layer 2
TP3	EX	C	tile	None	soil temperatures layer 3
TP4	EX	C	tile	None	soil temperatures layer 4
TP5	EX	C	tile	None	soil temperatures layer 5
TP6	EX	C	tile	None	soil temperatures layer 6
EMIS	EX	1	tile	None	surface emissivity
ALBVR	EX	1	tile	None	surface albedo visible beam
ALBVF	EX	1	tile	None	surface albedo visible diffuse

Short Name	Type	Units	Dims	Vert Loc	Long name
ALBNR	EX	1	tile	None	surface albedo near infrared beam
ALBNF	EX	1	tile	None	surface albedo near infrared diffuse
DELTS	EX	K	tile	None	change surface skin temperature
DELQS	EX	kg kg^{-1}	tile	None	change surface specific humidity
DELEVAP	EX	kg m^{-2} s^{-1}	tile	None	change evaporation
DELSH	EX	W m^{-2}	tile	None	change upward sensible energy flux
TST	EX	K	tile	None	surface skin temperature
LST	EX	K	tile	None	land surface skin temperature
QST	EX	kg kg^{-1}	tile	None	surface specific humidity
TH	EX	K	tile	None	turbulence surface skin temperature
QH	EX	kg kg^{-1}	tile	None	turbulence surface skin specific hum
CHT	EX	kg m^{-2} s^{-1}	tile	None	surface heat exchange coefficient
CMT	EX	kg m^{-2} s^{-1}	tile	None	surface momentum exchange coefficient
CQT	EX	kg m^{-2} s^{-1}	tile	None	surface moisture exchange coefficient
CNT	EX	1	tile	None	neutral drag coefficient
RIT	EX	1	tile	None	surface bulk richardson number
Z0	EX	m	tile	None	surface roughness
MOT2M	EX	K	tile	None	temperature
MOQ2M	EX	kg kg^{-1}	tile	None	humidity
MOU2M	EX	m s^{-1}	tile	None	zonal
MOV2M	EX	m s^{-1}	tile	None	meridional
MOT10M	EX	K	tile	None	temperature
MOQ10M	EX	kg kg^{-1}	tile	None	humidity
MOU10M	EX	m s^{-1}	tile	None	zonal
MOV10M	EX	m s^{-1}	tile	None	meridional
MOU50M	EX	m s^{-1}	tile	None	zonal
MOV50M	EX	m s^{-1}	tile	None	meridional
Z0H	EX	m	tile	None	surface roughness for heat
D0	EX	m	tile	None	zero plane displacement height
GUST	EX	m s^{-1}	tile	None	gustiness
VENT	EX	m s^{-1}	tile	None	surface ventilation velocity
ACCUM	EX	kg m^{-2} s^{-1}	tile	None	net ice accumulation rate
EVLAND	EX	kg m^{-2} s^{-1}	tile	None	Evaporation land

Short Name	Type	Units	Dims	Vert Loc	Long name
LHLAND	EX	W m^{-2}	tile	None	Latent heat flux land
SHLAND	EX	W m^{-2}	tile	None	Sensible heat flux land
SWLAND	EX	W m^{-2}	tile	None	Net shortwave land
SWDOWNLAND	EX	W m^{-2}	tile	None	Incident shortwave land
LWLAND	EX	W m^{-2}	tile	None	Net longwave land
GHLAND	EX	W m^{-2}	tile	None	Ground heating land
GHTSKIN	EX	W m^{-2}	tile	None	Ground heating skin temp
GHSNOW	EX	W m^{-2}	tile	None	Ground heating snow
SMLAND	EX	kg m^{-2} s^{-1}	tile	None	Snowmelt flux land
TWLAND	EX	kg m^{-2}	tile	None	Avail water storage land
TELAND	EX	J m^{-2}	tile	None	Total energy storage land
TSLAND	EX	kg m^{-2}	tile	None	Total snow storage land
DWLAND	EX	kg m^{-2} s^{-1}	tile	None	rate of change of total land water
DHLAND	EX	W m^{-2}	tile	None	rate of change of total land energy
SPLAND	EX	W m^{-2}	tile	None	rate of spurious land energy source
SPWATR	EX	kg m^{-2} s^{-1}	tile	None	rate of spurious land water source
SPSNOW	EX	W m^{-2}	tile	None	rate of spurious snow energy
ITY	EX	1	tile	None	vegetation type

13.2 RUN1 – First Run stage for the catchment component

INTERFACE:

```
subroutine RUN1 ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp),intent(inout) :: GC     !Gridded component
type(ESMF_State),   intent(inout) :: IMPORT !Import state
type(ESMF_State),   intent(inout) :: EXPORT !Export state
type(ESMF_Clock),   intent(inout) :: CLOCK  !The clock
integer,optional,   intent(out  ) :: RC     !Error code:
```

DESCRIPTION:
Does the cds computation and roughness length

14 Module GEOS_ChemEnvGridCompMod – Prepares Environment for GEOSchem

INTERFACE:

```
module GEOSChemEnvGridCompMod
```

USES:

```
use ESMF
use MAPL_Mod
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

DESCRIPTION:
This is a Cinderella gridded component (GC)

14.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer, intent(OUT)               :: RC  ! return code
```

DESCRIPTION:
The SetServices for the Chemistry Env GC needs to register its Initialize and Run. It uses the MAPL_Generic construct for defining state specs and couplings among its children. In addition, it creates the children GCs and runs their respective SetServices. *STATES:*

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
PLE	IM	Pa	xyz	Edge	air pressure
TH	IM	K	xyz	Center	potential temperature

Short Name	Type	Units	Dims	Vert Loc	Long name
Q	IM	kg kg^{-1}	xyz	Center	specific humidity
CN_PRCP	IM	kg m^{-2} s^{-1}	xy	None	Surface
TPREC	IM	kg m^{-2} s^{-1}	xy	None	total precipitation
FRLAND	IM	1	xy	None	fraction of land
FRLANDICE	IM	1	xy	None	fraction of land ice
FROCEAN	IM	1	xy	None	fraction of ocean
FRACI	IM	1	xy	None	ice covered fraction of tile
TS	IM	K	xy	None	TS
AIRDENS	EX	kg m^{-3}	xyz	Center	air density
DELP	EX	Pa	xyz	Center	pressure thickness
TPREC	EX	kg m^{-2} s^{-1}	xy	None	total precipitation
CN_PRCP	EX	kg m^{-2} s^{-1}	xy	None	Convective
NCN_PRCP	EX	kg m^{-2} s^{-1}	xy	None	Non-convective

15 Module GEOS_ChemGridCompMod – Parent Aerosol/-Chemistry Component

INTERFACE:

```
module GEOS_ChemGridCompMod
```

USES:

```
use ESMF
use MAPL_Mod
use Chem_Mod
use Chem_UtilMod

use GEOS_ChemEnvGridCompMod,  only : EChemSetServices    = > SetServices
use GOCART_GridCompMod,       only : AChemSetServices    = > SetServices
use StratChem_GridCompMod,    only : SChemSetServices    = > SetServices
use GMIchem_GridCompMod,      only : GChemSetServices    = > SetServices
use CARMAchem_GridCompMod,    only : CChemSetServices    = > SetServices
use GEOSCHEMchem_GridCompMod, only : GCChemSetServices   = > SetServices
use MATRIXchem_GridCompMod,   only : MChemSetServices    = > SetServices
use MAMchem_GridCompMod,      only : MAMChemSetServices  = > SetServices
use GEOS_PChemGridCompMod,    only : PChemSetServices    = > SetServices
use GAAS_GridCompMod,         only : GAASSetServices     = > SetServices
use H2O_GridCompMod,          only : H2OSetServices      = > SetServices
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices

                                              -----------

Private state
-------------
TYPE GEOS_ChemGridComp
   PRIVATE
   type(ESMF_Config), pointer :: CF  ! Private Config
   LOGICAL :: enable_PCHEM
   LOGICAL :: enable_GOCART
   LOGICAL :: enable_GAAS
   LOGICAL :: enable_H2O
   LOGICAL :: enable_STRATCHEM
   LOGICAL :: enable_GMICHEM
   LOGICAL :: enable_CARMA
   LOGICAL :: enable_GEOSCHEM
   LOGICAL :: enable_MATRIX
   LOGICAL :: enable_MAM
   INTEGER :: AERO_PROVIDER
   INTEGER :: RATS_PROVIDER          ! WARNING: May be multiple RATS_PROVIDERs
END TYPE GEOS_ChemGridComp

Hook for the ESMF
-----------------
TYPE GEOS_ChemGridComp_Wrap
   TYPE (GEOS_ChemGridComp), pointer :: PTR  = > null()
END TYPE GEOS_ChemGridComp_Wrap

= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
```

DESCRIPTION:
This gridded component (GC) combines

15.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer,             intent(  OUT) :: RC  ! return code
```

DESCRIPTION:
The SetServices for the Chemistry GC needs to register its Initialize and Run. It uses the MAPL_Generic construct for defining state specs and couplings among its children. In addition, it creates the children GCs and runs their respective SetServices.

15.2 Initialize – Initialized method for composite Aero-Chemistry

INTERFACE:

```
subroutine Init ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code
```

DESCRIPTION:
The Initialize method of the Chemistry Composite Gridded Component. It acts as a driver for the initializtion of the children.

16 Module GEOS_DataSea – A 'fake' ocean surface

INTERFACE:

```
module GEOSDataSeaGridCompMod
```

USES:

```
use ESMF
use MAPL_Mod
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

DESCRIPTION:
`GEOS_DataSea` is a gridded component that reads the ocean_bcs file This module interpolates the SST and sea ice data from either daily or monthly values to the correct time of the simulation. Data are read only if the simulation time is not in the save interval. Surface Albedo and Surface roughness calculations are also takencare of in this module.

16.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer, optional                  :: RC  ! return code
```

DESCRIPTION:
This version uses the MAPL_GenericSetServices. This function sets the Initialize and Finalize services, as well as allocating our instance of a generic state and putting it in the gridded component (GC). Here we only need to set the run method and add the state variable specifications (also generic) to our instance of the generic state. This is the way our true state variables get into the ESMF_State INTERNAL, which is in the MAPL_MetaComp.
STATES:

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
HW	IM	m	xy	None	water skin layer depth
TW	IM	K	xy	None	water skin temperature
SW	IM	psu	xy	None	water skin salinity
SWHEAT	IM	W m^{-2}	xyz	Center	solar heating rate
UW	EX	m s^{-1}	xy	None	zonal velocity of surface water
VW	EX	m s^{-1}	xy	None	meridional velocity of surface water
TS_FOUND	EX	K	xy	None	foundation temperature for interface layer

---

16.2 RUN – Run stage for the DataSea component

INTERFACE:

```
subroutine RUN ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code:
```

DESCRIPTION:
Periodically refreshes the SST and Ice information.

17 Module GEOS_DataSeaIce – A 'fake' seaice model

INTERFACE:

```
module GEOSDataSeaIceGridCompMod
```

USES:

```
use ESMF
use MAPL_Mod
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

DESCRIPTION:
`GEOS_DataSeaIce` is a gridded component that reads the ocean_bcs file This module interpolates the SST and sea ice fraction data from either daily or monthly values to the correct time of the simulation. Data are read only if the simulation time is not in the save interval. Surface Albedo and Surface roughness calculations are also takencare of in this module.

17.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer, optional                  :: RC  ! return code
```

DESCRIPTION:
This version uses the MAPL_GenericSetServices. This function sets the Initialize and Finalize services, as well as allocating our instance of a generic state and putting it in the gridded component (GC). Here we only need to set the run method and add the state variable specifications (also generic) to our instance of the generic state. This is the way our true state variables get into the ESMF_State INTERNAL, which is in the MAPL_MetaComp.
STATES:

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
HI	IM	m	xy	None	seaice skin layer depth
TI	IM	K	xy	None	seaice skin temperature
SI	IM	psu	xy	None	seaice skin salinity
HW	IM	kg m^{-2}	xy	None	water skin layer mass
TW	IM	K	xy	None	water skin temperature
SW	IM	psu	xy	None	water skin salinity
UI	EX	m s^{-1}	xy	None	zonal velocity of surface seaice
VI	EX	m s^{-1}	xy	None	meridional velocity of surface seaice
FRACICE	EX	1	xy	None	fractional cover of seaice
MELTQ	EX	W m^{-2}	xy	None	heat of melting or freezing

17.2 RUN – Run stage for the DataSeaIce component

INTERFACE:

```
subroutine RUN ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code:
```

DESCRIPTION:
Periodically refreshes the SST and Ice information.

18 Module GEOS_Singcol – A Module to drive single column model with profile data.

USES:

```
use ESMF
use MAPL_Mod
use PPM
use cfmip_data_mod
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

DESCRIPTION:
MOIST

18.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
  type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
  integer,             intent(  OUT) :: RC  ! return code
```

DESCRIPTION:
This version uses the GEOS_GenericSetServices. This function sets the Initialize and Finalize services, as well as allocating our instance of a generic state and putting it in the gridded component (GC). Here we only need to set the run method and add the state variable specifications (also generic) to our instance of the generic state. This is the way our true state variables get into the ESMF_State INTERNAL, which is in the GEOS_GenericState.
STATES:

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
$\overline{\text{PREF}}$	IN	Pa	z	Edge	reference air pressure
PLE	IN	Pascals	xyz	Edge	Pressure at the edges
T	IN	K	xyz	Center	air temperature
U	IN	m/s	xyz	Center	Zonal
V	IN	m/s	xyz	Center	meridional
OM	IN	Pa/s	xyz	Edge	pressure
DTDT	IM	K s^{-1}	xyz	Center	T tendency
DUDT	IM	m s^{-1} s^{-1}	xyz	Center	later
DVDT	IM	m s^{-1} s^{-1}	xyz	Center	later
DPEDT	IM	Pa	xyz	Edge	air pressure
DQVANA	IM	kg kg^{-1}	xyz	Center	specific humidity increment from analysis
DOXANA	IM	kg kg^{-1}	xyz	Center	ozone increment from analysis
PHIS	IM	m^2 sec^{-2}	xy	None	surface geopotential height
TRADV	IM	unknown			advected quantities
PLE	EX	Pascals	xyz	Edge	Pressure
T	EX	K	xyz	Center	air temperature
U	EX	m/s	xyz	Center	Zonal
V	EX	m/s	xyz	Center	meridional
Q	EX	m/s	xyz	Center	meridional
OMEGA	EX	Pa/s	xyz	Edge	pressure
DIV	EX	s^{-1}	xyz	Center	divergence
LHOBS	EX	W m^{-2}	xy	None	Obs.
SHOBS	EX	W m^{-2}	xy	None	Obs.
PCPOBS	EX	mm/d	xy	None	Obs.
TSAIROBS	EX	K	xy	None	Obs.
TGSOILOBS	EX	K	xy	None	Obs.
PSFCOBS	EX	[PahPa]	xy	None	Obs.
TH	EX	K	xyz	Center	potential temperature
ZLE	EX	m	xyz	Edge	Geop.
QOBS	EX	1	xyz	Center	Obs.
TOBS	EX	K	xyz	Center	Obs.

Short Name	Type	Units	Dims	Vert Loc	Long name
TVATOBS	EX	K s^{-1}	xyz	Center	Obs.
THATOBS	EX	K s^{-1}	xyz	Center	Obs.
QVATOBS	EX	K s^{-1}	xyz	Center	Obs.
QHATOBS	EX	K s^{-1}	xyz	Center	Obs.
QTEST	EX	1	xyz	Center	test tracer
UE	EX	m s^{-1}	xyz	Center	Diagnosed Edge Winds
WWTG	EX	m s^{-1}	xyz	Edge	weak T-gradient compensating W
WTOT	EX	m s^{-1}	xyz	Edge	total vertical velocity
QLOBS	EX	1	xyz	Center	Obs.
QIOBS	EX	1	xyz	Center	Obs.
PS	EX	Pa	xy	None	Surface
TA	EX	K	xy	None	Surface air temperature
SPEED	EX	m s^{-1}	xy	None	Surface wind speed
DZ	EX	m	xy	None	Surface layer height
QA	EX	1	xy	None	Surface air spec humidity
QSKINOBS	EX	1	xy	None	obs skin spec humidity whatever that means
TSKINOBS	EX	K	xy	None	obs skin temperature
PHIS	EX	K	xy	None	obs skin temperature
VARFLT	EX	K	xy	None	obs skin temperature
PEANA	EX	W m^{-2}	xy	None	total potential energy tendency due to analysis
PEPHY	EX	W m^{-2}	xy	None	total potential energy tendency due to physics
DOXDTANAINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated ozone tendency due to analysis
DQVDTANAINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated water vapor tendency due to analysis
DQLDTANAINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated liquid water tendency due to analysis
DQIDTANAINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated ice water tendency due to analysis
DTHVDTANAINT	EX	K kg m^{-2} s^{-1}	xy	None	vertically integrated THV tendency due to analysis
DTHVDTPHYINT	EX	K kg m^{-2} s^{-1}	xy	None	vertically integrated THV tendency due to physics
DQVDTDYNINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated water vapor tendency due to dynamics
PV	EX	m^2 kg^{-1} sec^{-1}	xyz	Center	ertels isentropic potential vorticity
EPV	EX	K m^2 kg^{-1} sec^{-1}	xyz	Center	ertels potential vorticity

Short Name	Type	Units	Dims	Vert Loc	Long name
TROPP_BLENDED	EX	Pa	xy	None	tropopause pressure based on blended estimate
S	EX	J kg^{-1}	xyz	Center	static energy
TV	EX	K	xyz	Center	air virtual temperature
PLK	EX	J kg^{-1}	xyz	Center	Exner quantity
PL	EX	Pa	xyz	Center	midlevel pressures
DQVDTDYN	EX	kg/kg/sec	xyz	Center	tendency of specific humidity due to dynamics
DQLLSDTDYN	EX	kg/kg/sec	xyz	Center	tendency of large scale cloud water due to dynamics
DQILSDTDYN	EX	kg/kg/sec	xyz	Center	tendency of large scale cloud ice due to dynamics
DQLCNDTDYN	EX	kg/kg/sec	xyz	Center	tendency of convective cloud water due to dynamics
DQICNDTDYN	EX	kg/kg/sec	xyz	Center	tendency of convective cloud ice due to dynamics
DCLLSDTDYN	EX	fraction	xyz	Center	tendency of large scale cloud fraction due to dynamics
DCLCNDTDYN	EX	fraction	xyz	Center	tendency of convective cloud fraction due to dynamics
DTDTDYN	EX	K sec^{-1}	xyz	Center	tendency of air temperature due to dynamics
HDQDTDYN	EX	kg/kg/sec	xyz	Center	horiz tendency of specific humidity due to dynamics
VDQDTDYN	EX	kg/kg/sec	xyz	Center	vertical tendency of specific humidity due to dynamics
HDTDTDYN	EX	K sec^{-1}	xyz	Center	horiz tendency of air temperature due to dynamics
HDTHDTDYN	EX	K sec^{-1}	xyz	Center	horiz tendency of air pot temp due to dynamics
VDTDTDYN	EX	K sec^{-1}	xyz	Center	vertical tendency of air temperature due to dynamics
VDTHDTDYN	EX	K sec^{-1}	xyz	Center	vertical tendency of air pot temp due to dynamics
AREA	EX	m^2	xy	None	agrid cell area
AK	EX	1	z	Edge	hybrid sigma pressure a
BK	EX	1	z	Edge	hybrid sigma pressure b
U_CGRID	EX	m s^{-1}	xyz	Center	eastward wind on C-Grid
V_CGRID	EX	m s^{-1}	xyz	Center	northward wind on C-Grid
U_DGRID	EX	m s^{-1}	xyz	Center	eastward wind on native D-Grid
V_DGRID	EX	m s^{-1}	xyz	Center	northward wind on native D-Grid
PT	EX	K $Pa^{-\kappa}$	xyz	Center	scaled potential temperature

Short Name	Type	Units	Dims	Vert Loc	Long name
PE	EX	Pa	xyz	Edge	air pressure

18.2 RUN – Run method for the SINGLE COLUMN component

INTERFACE:

```
subroutine RUN ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code:
```

DESCRIPTION:
This version uses the GEOS_GenericSetServices. This function sets the Initialize and Finalize services, as well as allocating

19 Module GEOS_GcmGridCompMod – A Module to combine Agcm and Ogcm Gridded Components

INTERFACE:

```
module GEOSGcmGridCompMod
```

USES:

```
 use ESMF
 use MAPL_Mod

 use GEOS_dataatmGridCompMod, only:  DATAATM_SetServices  = > SetServices
use GEOS_AgcmGridCompMod,     only:  AGCM_SetServices  = > SetServices
use GEOS_mkiauGridCompMod,    only:  AIAU_SetServices  = > SetServices
use DFI_GridCompMod,          only:  ADFI_SetServices  = > SetServices
use GEOS_OgcmGridCompMod,     only:  OGCM_SetServices  = > SetServices


  integer :: chdir, flag  =  0
  external chdir
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices

integer, parameter :: NUM_ICE_CATEGORIES = 5
integer, parameter :: NUM_ICE_LAYERS = 4
integer, parameter :: NUM_SNOW_LAYERS = 1


= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
```

DESCRIPTION:
This gridded component (GC) combines the Agcm GC, and Ogcm GC into a new composite Gcm GC.

19.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer, optional                  :: RC  ! return code
```

DESCRIPTION:
The SetServices for the PhysicsGcm GC needs to register its Initialize and Run. It uses the MAPL_Generic construct for defining state specs and couplings among its children. In addition, it creates the children GCs (AGCM and OGCM) and runs their respective SetServices.

20 Module GEOS_Gwd – A Module to compute the forcing due to parameterized gravity wave drag

DESCRIPTION:
`GWD` is a light-weight gridded component to compute the forcing due to gravity wave drags. It operates on the ESMF grid that appears in the gridded component passed to its `Initialize` method. Unlike heavier gridded components, it does not enforce its own grid. The only restrictions are that it be a 3-dimensional grid in which one dimension is aligned with the vertical coordinate and only the horizontal dimensions are decomposed.
The gravity wave drag scheme is based on NCAR WACCM1b gw_drag routine. The scheme includes parameterizations for orographic (stationary) gravity waves (Kiehl et al. 1996), and

for a spectrum of traveling gravity waves (Sassi et al. 2003; http://acd.ucar.edu/models/WACCM). Both parameteriz- ations are based on Lindzen's [1981] formulation. The interested reader is referred to those publications for details of the mathematical derivations.

USES:

```
use ESMF
use MAPL_Mod

use gw_drag, only: &
      ! Subroutines
      GW_INTR, &
      ! Working Arrays
      ZM_DEV, LNPINT_DEV, PMLN_DEV, PMID_DEV, &
      RPDEL_DEV, EFFGWBKG_DEV, EFFGWORO_DEV, &
      ! Inputs - PREGEO
      PINT_DEV, RLAT_DEV, &
      ! Outputs - PREGEO
      PDEL_DEV, &
      ! Inputs - GEOPOT
      T_DEV, Q_DEV, &
      ! Outputs - GEOPOT
      ZI_DEV, &
      ! Inputs - INTR
      U_DEV, V_DEV, SGH_DEV, PREF_DEV, &
      ! Outputs - INTR
      DUDT_GWD_DEV, DVDT_GWD_DEV, DTDT_GWD_DEV, &
      DUDT_ORG_DEV, DVDT_ORG_DEV, DTDT_ORG_DEV, &
      TAUGWDX_DEV, TAUGWDY_DEV, TAUOX_DEV, TAUOY_DEV, &
      FEO_DEV, FEPO_DEV,  TAUBKGX_DEV, TAUBKGY_DEV, &
      TAUBX_DEV, TAUBY_DEV,  FEB_DEV, FEPB_DEV, &
      UTBSRC_DEV, VTBSRC_DEV, TTBSRC_DEV, &
      ! Outputs - POSTINTR
      DUDT_TOT_DEV, DVDT_TOT_DEV, DTDT_TOT_DEV, &
      DUDT_RAH_DEV, DVDT_RAH_DEV, DTDT_RAH_DEV, &
      PEGWD_DEV, PEORO_DEV, PERAY_DEV, PEBKG_DEV, &
      KEGWD_DEV, KEORO_DEV, KERAY_DEV, KEBKG_DEV, &
      KERES_DEV, BKGERR_DEV
use cudafor
use gw_drag, only: gw_intr
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

20.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer, optional                  :: RC  ! return code
```

DESCRIPTION:
This version uses the MAPL_GenericSetServices. This function sets the Initialize and Finalize services, as well as allocating our instance of a generic state and putting it in the gridded component (GC). Here we only need to set the run method and add the state variable specifications (also generic) to our instance of the generic state. This is the way our true state variables get into the ESMF_State INTERNAL, which is in the MAPL_MetaComp.
STATES:

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
PLE	IM	Pa	xyz	Edge	air pressure
T	IM	K	xyz	Center	air temperature
Q	IM	kg kg^{-1}	xyz	Center	specific humidity
U	IM	m s^{-1}	xyz	Center	eastward wind
V	IM	m s^{-1}	xyz	Center	northward wind
SGH	IM	m	xy	None	standard deviation of topography
PREF	IM	Pa	z	Edge	reference air pressure
QI	IM	kg kg^{-1}	xyz	Center	specific humidity of suspended ice
LS_PRCP	IM	kg m^{-2} s^{-1}	xy	None	Total LS Surface precipitation flux
PLE	EX	Pa	xyz	Edge	air pressure
T	EX	K	xyz	Center	air temperature
Q	EX	kg kg^{-1}	xyz	Center	specific humidity
U	EX	m s^{-1}	xyz	Center	eastward wind
V	EX	m s^{-1}	xyz	Center	northward wind
SGH	EX	m	xy	None	standard deviation of topography
PREF	EX	Pa	z	Edge	reference air pressure
DTDT	EX	Pa K s^{-1}	xyz	Center	mass weighted air temperature tendency due to GWD

Short Name	Type	Units	Dims	Vert Loc	Long name
TTMGW	EX	K s^{-1}	xyz	Center	air temperature tendency due to GWD
DUDT	EX	m s^{-2}	xyz	Center	tendency of eastward wind due to GWD
DVDT	EX	m s^{-2}	xyz	Center	tendency of northward wind due to GWD
DTDT_ORO	EX	K s^{-1}	xyz	Center	air temperature tendency due to orographic GWD
DUDT_ORO	EX	m s^{-2}	xyz	Center	tendency of eastward wind due to orographic GWD
DVDT_ORO	EX	m s^{-2}	xyz	Center	tendency of northward wind due to orographic GWD
DTDT_BKG	EX	K s^{-1}	xyz	Center	air temperature tendency due to background GWD
DUDT_BKG	EX	m s^{-2}	xyz	Center	tendency of eastward wind due to background GWD
DVDT_BKG	EX	m s^{-2}	xyz	Center	tendency of northward wind due to background GWD
DTDT_RAY	EX	K s^{-1}	xyz	Center	air temperature tendency due to Rayleigh friction
DUDT_RAY	EX	m s^{-2}	xyz	Center	tendency of eastward wind due to Rayleigh friction
DVDT_RAY	EX	m s^{-2}	xyz	Center	tendency of northward wind due to Rayleigh friction
TAUGWX	EX	N m^{-2}	xy	None	surface eastward gravity wave stress
TAUGWY	EX	N m^{-2}	xy	None	surface northward gravity wave stress
TAUOROX	EX	N m^{-2}	xy	None	surface eastward orographic gravity wave stress
TAUOROY	EX	N m^{-2}	xy	None	surface northward orographic gravity wave stress
TAUBKGX	EX	N m^{-2}	xy	None	surface eastward background gravity wave stress
TAUBKGY	EX	N m^{-2}	xy	None	surface northward background gravity wave stress
TAUMSTX	EX	N m^{-2}	xy	None	surface eastward gravity wave stress due to Moist Processes
TAUMSTY	EX	N m^{-2}	xy	None	surface northward gravity wave stress due to Moist Processes
CLDSTD	EX	m	xy	None	gravity wave drag standard deviation due to clouds

Short Name	Type	Units	Dims	Vert Loc	Long name
UBASE	EX	m s^{-1}	xy	None	eastward component of base level wind
VBASE	EX	m s^{-1}	xy	None	northward component of base level wind
UBAR	EX	m s^{-1}	xy	None	eastward component of mean level wind
VBAR	EX	m s^{-1}	xy	None	northward component of mean level wind
PEGWD	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency across gwd
PEORO	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency due to orographic gravity waves
PEBKG	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency due to gravity wave background
PERAY	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency due to Rayleigh friction
KEGWD	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency across gwd
KEORO	EX	W m^{-2}	xy	None	vertically integrated kinetic energy dissipation due to orographic gravity waves
KERAY	EX	W m^{-2}	xy	None	vertically integrated kinetic energy dissipation due to Rayleigh friction
KEBKG	EX	W m^{-2}	xy	None	vertically integrated kinetic energy dissipation due to gravity wave background
KERES	EX	W m^{-2}	xy	None	vertically integrated kinetic energy residual for total energy conservation
BKGERR	EX	W m^{-2}	xy	None	vertically integrated kinetic energy residual for BKG energy conservation

20.2 RUN – Run method for the GWD component

INTERFACE:

```
subroutine RUN ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code:
```

DESCRIPTION:
This version uses the MAPL_GenericSetServices. This function sets the Initialize and Finalize services, as well as allocating

21 Module GEOS_Irrad – A Module to compute longwaves radiative transfer through a cloudy atmosphere

DESCRIPTION:
`Irrad` is a light-weight gridded component to compute longwave radiative fluxes. It operates on the ESMF grid that appears in the gridded component passed to its `Initialize` method. Unlike heavier gridded components, it does not enforce its own grid. The only restrictions are that it be a 3-dimensional grid in which one dimension is aligned with the vertical coordinate and only the horizontal dimensions are decomposed.
The radiative transfer calculation is based on M-D Chou's IRRAD routine. A full documentation of the code may be found in "A Thermal Infrared Radiation Parameterization for Atmospheric Studies" M.-D. Chou et al., NASA/TM-2001-104606, Vol. 19, 55 pp, 2003. Based on the 1996-version of the Air Force Geophysical Laboratory HITRAN data base (Rothman et al., 1998), the parameterization includes the absorption due to major gaseous absorption (water vapor, CO2 , O3) and most of the minor trace gases (N2O, CH4 , CFC's), as well as clouds and aerosols. The thermal infrared spectrum is divided into nine bands and a subband. To achieve a high degree of accuracy and speed, various approaches of computing the transmission function are applied to different spectral bands and gases. The gaseous transmission function is computed either using the k-distribution method or the table look-up method. To include the effect of scattering due to clouds and aerosols, the optical thickness is scaled by the single-scattering albedo and asymmetry factor. The optical thickness, the single-scattering albedo, and the asymmetry factor of clouds are parameterized as functions of the ice and water content and the particle size.
All outputs are optional and are filled only if they have been initialized by a coupler.
The net (+ve downward) fluxes are returned at the layer interfaces, which are indexed from the top of the atmosphere (L = 0) to the surface. It also computes the sensitivity of net downward flux to surface temperature and emission by the surface. The full transfer calculation, including the linearization w.r.t. the surface temperature, is done intermitently, on the component's main time step and its results are kept in the internal state. Exports are refreshed each heartbeat based on the latest surface temperature.

Radiation should be called either before or after thos components (usually SURFACE and DYNAMICS) that use its fluxes and modify its inputs. If it is called before, the intemittent refresh should occur during the first step of the radiation cycle, while if it is called after, it should occur during the last step. The behavior of the component needs to be somewhat different in these two cases and so a means is provided, through the logical attribute CALL_LAST in configuration, of telling the component how it is being used. The default is CALL_LAST = "TRUE".

USES:

```
use ESMF
use MAPL_Mod
use GEOS_UtilsMod
use AeroOptPropTableMod

use rrtmg_lw_rad, only: rrtmg_lw  !  RRTMG Code
use mcica_subcol_gen_lw, only: mcica_subcol_lw

use cudafor
! NOTE: USE renames are used below to prevent name clashes with
!       CUDA copies to the GPU.
use rad_constants, only: &
      AIB_IR_CONST = >AIB_IR, AWB_IR_CONST = >AWB_IR, &
      AIW_IR_CONST = >AIW_IR, AWW_IR_CONST = >AWW_IR, &
      AIG_IR_CONST = >AIG_IR, AWG_IR_CONST = >AWG_IR
use irrad_constants, only: &
      XKW_CONST = >XKW, XKE_CONST = >XKE,  MW_CONST = >MW,  &
       AW_CONST = >AW,   BW_CONST = >BW,                   &
       PM_CONST = >PM,  FKW_CONST = >FKW, GKW_CONST = >GKW, &
       CB_CONST = >CB,  DCB_CONST = >DCB,                  &
      W11_CONST = >W11, W12_CONST = >W12, W13_CONST = >W13, &
      P11_CONST = >P11, P12_CONST = >P12, P13_CONST = >P13, &
      DWE_CONST = >DWE, DPE_CONST = >DPE,                  &
       C1_CONST = >C1,   C2_CONST = >C2,   C3_CONST = >C3,  &
      OO1_CONST = >OO1, OO2_CONST = >OO2, OO3_CONST = >OO3, &
      H11_CONST = >H11, H12_CONST = >H12, H13_CONST = >H13, &
      H21_CONST = >H21, H22_CONST = >H22, H23_CONST = >H23, &
      H81_CONST = >H81, H82_CONST = >H82, H83_CONST = >H83
use irradmod, only: &
      ! Subroutines
      IRRAD, &
      ! Parameters
      NX, NO, NC, NH, &
      ! Inputs
      PLE_DEV, TA_DEV, WA_DEV, OA_DEV, TB_DEV, &
      N2O_DEV, CH4_DEV, CFC11_DEV, CFC12_DEV, CFC22_DEV, &
      FS_DEV, TG_DEV, EG_DEV, TV_DEV, EV_DEV, &
```

```
        RV_DEV, CWC_DEV, FCLD_DEV, REFF_DEV, &
        ! Aerosol inputs
        TAUA_DEV, SSAA_DEV, ASYA_DEV, &
        ! Constant arrays in global memory
         C1,  C2,  C3, &
        OO1, OO2, OO3, &
        H11, H12, H13, &
        H21, H22, H23, &
        H81, H82, H83, &
        ! Outputs
        FLXU_DEV, FLCU_DEV, FLAU_DEV, &
        FLXD_DEV, FLCD_DEV, FLAD_DEV, &
        DFDTS_DEV, SFCEM_DEV, TAUDIAG_DEV, &
        ! Constants
        XKW, XKE, MW, AW, BW, PM, FKW, &
        GKW, AIB_IR, AWB_IR, AIW_IR, AWW_IR, AIG_IR, AWG_IR, &
        CB, DCB, W11, W12, W13, P11, P12, &
        P13, DWE, DPE
  use irradmod, only: IRRAD
```

PUBLIC MEMBER FUNCTIONS:

```
  public SetServices
```

21.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
  subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
    type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
    integer, optional                  :: RC  ! return code
```

DESCRIPTION:
This version uses the MAPL_GenericSetServices. This function sets the Initialize and Finalize services, as well as allocating our instance of a generic state and putting it in the gridded component (GC). Here we only need to set the run method and add the state variable specifications (also generic) to our instance of the generic state. This is the way our true state variables get into the ESMF_State INTERNAL, which is in the MAPL_MetaComp.

STATES:

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
PLE	IM	Pa	xyz	Edge	air pressure
T	IM	K	xyz	Center	air temperature
QV	IM	kg kg^{-1}	xyz	Center	specific humidity
QL	IM	kg kg^{-1}	xyz	Center	mass fraction of cloud liquid water in air
QI	IM	kg kg^{-1}	xyz	Center	mass fraction of cloud ice in air
QR	IM	kg kg^{-1}	xyz	Center	mass fraction of rain water in air
QS	IM	kg kg^{-1}	xyz	Center	mass fraction of snow in air
RL	IM	m	xyz	Center	effective radius of cloud liquid water particles
RI	IM	m	xyz	Center	effective radius of cloud ice particles
RR	IM	m	xyz	Center	effective radius of rain particles
RS	IM	m	xyz	Center	effective radius of snow particles
O3	IM	kg kg^{-1}	xyz	Center	ozone mass mixing ratio
CH4	IM	pppv	xyz	Center	methane concentration
N2O	IM	pppv	xyz	Center	nitrous oxide concentration
CFC11	IM	pppv	xyz	Center	CFC11 concentration
CFC12	IM	pppv	xyz	Center	CFC12 concentration
HCFC22	IM	pppv	xyz	Center	HCFC22 concentration
FCLD	IM	1	xyz	Center	cloud area fraction in atmosphere layer
TS	IM	K	xy	None	surface skin temperature
EMIS	IM	1	xy	None	surface emissivity
PREF	IM	Pa	z	Edge	reference air pressure
TSINST	IM	K	xy	None	surface skin temperature
AERO	IM	1	xyz	Center	aerosols
FLX	EX	W m^{-2}	xyz	Edge	net downward longwave flux in air
FLXD	EX	W m^{-2}	xyz	Edge	downward longwave flux in air
FLXU	EX	W m^{-2}	xyz	Edge	upward longwave flux in air
FLC	EX	W m^{-2}	xyz	Edge	net downward longwave flux in air assuming clear sky

Short Name	Type	Units	Dims	Vert Loc	Long name
FLCD	EX	W m^{-2}	xyz	Edge	downward longwave flux in air assuming clear sky
FLCU	EX	W m^{-2}	xyz	Edge	upward longwave flux in air assuming clear sky
FLA	EX	W m^{-2}	xyz	Edge	net downward longwave flux in air assuming clear sky and no aerosol
FLAD	EX	W m^{-2}	xyz	Edge	downward longwave flux in air assuming clear sky and no aerosol
FLAU	EX	W m^{-2}	xyz	Edge	upward longwave flux in air assuming clear sky and no aerosol
SFCEM	EX	W m^{-2}	xy	None	longwave flux emitted from surface
SFCEM0	EX	W m^{-2}	xy	None	longwave flux emitted from surface at reference time
LWS0	EX	W m^{-2}	xy	None	surface absorbed longwave radiation at reference time
DSFDTS	EX	W m^{-2} K^{-1}	xy	None	sensitivity of longwave flux emitted from surface to surface temperature
DSFDTS0	EX	W m^{-2} K^{-1}	xy	None	sensitivity of longwave flux emitted from surface to surface temperature at reference time
TSREFF	EX	K	xy	None	surface temperature
OLR	EX	W m^{-2}	xy	None	upwelling longwave flux at toa
OLC	EX	W m^{-2}	xy	None	upwelling longwave flux at toa assuming clear sky
OLA	EX	W m^{-2}	xy	None	upwelling longwave flux at toa assuming clear sky and no aerosol
FLNS	EX	W m^{-2}	xy	None	surface net downward longwave flux
FLNSC	EX	W m^{-2}	xy	None	surface net downward longwave flux assuming clear sky
FLNSA	EX	W m^{-2}	xy	None	surface net downward longwave flux assuming clear sky and no aerosol
LWS	EX	W m^{-2}	xy	None	surface absorbed longwave radiation

Short Name	Type	Units	Dims	Vert Loc	Long name
LCS	EX	W m^{-2}	xy	None	surface absorbed longwave radiation assuming clear sky
LAS	EX	W m^{-2}	xy	None	surface absorbed longwave radiation assuming clear sky and no aerosol
CLDTMP	EX	K	xy	None	cloud top temperature
CLDPRS	EX	Pa	xy	None	cloud top pressure
TAUIR	EX	W m^{-2}	xyz	Center	longwave cloud optical thickness at 800 cm-1
CLDTTLW	EX	1	xy	None	total cloud area fraction lw
CLDHILW	EX	1	xy	None	total cloud area fraction lw
CLDMDLW	EX	1	xy	None	total cloud area fraction lw
CLDLOLW	EX	1	xy	None	total cloud area fraction lw
FLX	IN	W m^{-2}	xyz	Edge	net downward longwave flux in air
FLC	IN	W m^{-2}	xyz	Edge	net downward longwave flux in air for clear sky(INTERNAL)
FLA	IN	W m^{-2}	xyz	Edge	net downward longwave flux in air for clear sky and no aerosol
FLXD	IN	W m^{-2}	xyz	Edge	downward longwave flux in air
FLXU	IN	W m^{-2}	xyz	Edge	upward longwave flux in air
FLCD	IN	W m^{-2}	xyz	Edge	downward longwave flux in air for clear sky
FLCU	IN	W m^{-2}	xyz	Edge	upward longwave flux in air for clear sky
FLAD	IN	W m^{-2}	xyz	Edge	downward longwave flux in air for clear sky and no aerosol
FLAU	IN	W m^{-2}	xyz	Edge	upward longwave flux in air for clear sky and no aerosol
DFDTS	IN	W m^{-2} K^{-1}	xyz	Edge	sensitivity of net downward longwave flux in air to surface temperature
DFDTSC	IN	W m^{-2} K^{-1}	xyz	Edge	sensitivity of net downward longwave flux in air to surface temperature for clear sky
SFCEM	IN	W m^{-2}	xy	None	longwave flux emitted from surface
TS	IN	K	xy	None	surface temperature

21.2 RUN – Run method for the LW component

INTERFACE:

```
subroutine RUN ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code:
```

DESCRIPTION:
Periodically refreshes the fluxes and their derivatives w.r.t surface skin temperature. On every step it produces a linear estimate of the fluxes based on the instantaneous surface temperature.

22 Module GEOS_LakeGridCompMod – Implements slab lake tiles.

USES:

```
use sfclayer  ! using module that contains sfc layer code
use ESMF
use MAPL_Mod
use GEOS_UtilsMod
use DragCoefficientsMod


integer, parameter :: ICE    =  1
integer, parameter :: WATER  =  2
integer, parameter :: NUM_SUBTILES  =  2
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

DESCRIPTION:
`GEOS_Lake` is a light-weight gridded component that updates the lake tiles

22.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer, optional                  :: RC  ! return code
```

DESCRIPTION:
This version uses the MAPL_GenericSetServices, which sets the Initialize and Finalize services, as well as allocating our instance of a generic state and putting it in the gridded component (GC). Here we only need to set the run method and add the state variable specifications (also generic) to our instance of the generic state. This is the way our true state variables get into the ESMF_State INTERNAL, which is in the MAPL_MetaComp. The import and internal variables are allocated and initialized by generic. Here generic is used for tiles. *STATES:*

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
EMIS	EX	1	tile	None	surface emissivity
ALBVR	EX	1	tile	None	surface albedo for visible beam
ALBVF	EX	1	tile	None	surface albedo for visible diffuse
ALBNR	EX	1	tile	None	surface albedo for near infrared beam
ALBNF	EX	1	tile	None	surface albedo for near infrared diffuse
EVAPOUT	EX	kg m^{-2} s^{-1}	tile	None	evaporation
SUBLIM	EX	kg m^{-2} s^{-1}	tile	None	sublimation
RUNOFF	EX	kg m^{-2} s^{-1}	tile	None	runoff flux
SHOUT	EX	W m^{-2}	tile	None	upward sensible heat flux
HLWUP	EX	W m^{-2}	tile	None	surface outgoing longwave flux
LWNDSRF	EX	W m^{-2}	tile	None	surface net downward longwave flux
SWNDSRF	EX	W m^{-2}	tile	None	surface net downward shortwave flux
HLATN	EX	W m^{-2}	tile	None	total latent energy flux
TST	EX	K	tile	None	surface skin temperature

Short Name	Type	Units	Dims	Vert Loc	Long name
QST	EX	kg kg^{-1}	tile	None	surface specific humidity
TH	EX	K	tile	None	turbulence surface skin temperature
QH	EX	kg kg^{-1}	tile	None	turbulence surface specific humidity
UH	EX	m s^{-1}	tile	None	turbulence surface zonal velocity
VH	EX	m s^{-1}	tile	None	turbulence surface meridional velocity
DELTS	EX	K	tile	None	change of surface skin temperature
DELQS	EX	kg kg^{-1}	tile	None	change of surface specific humidity
CHT	EX	kg m^{-2} s^{-1}	tile	None	surface heat exchange coefficient
CMT	EX	kg m^{-2} s^{-1}	tile	None	surface momentum exchange coefficient
CQT	EX	kg m^{-2} s^{-1}	tile	None	surface moisture exchange coefficient
CNT	EX	1	tile	None	neutral drag coefficient
RIT	EX	1	tile	None	surface bulk richardson number
PS	EX	Pa	tile	None	surface pressure
GUST	EX	m s^{-1}	tile	None	gustiness
VENT	EX	m s^{-1}	tile	None	surface ventilation velocity
Z0	EX	m	tile	None	surface roughness
Z0H	EX	m	tile	None	surface roughness for heat
MOT10M	EX	K	tile	None	temperature
MOQ10M	EX	kg kg^{-1}	tile	None	humidity
MOU10M	EX	m s^{-1}	tile	None	zonal
MOV10M	EX	m s^{-1}	tile	None	meridional
MOT2M	EX	K	tile	None	temperature
MOQ2M	EX	kg kg^{-1}	tile	None	humidity
MOU2M	EX	m s^{-1}	tile	None	zonal
MOV2M	EX	m s^{-1}	tile	None	meridional
MOU50M	EX	m s^{-1}	tile	None	zonal
MOV50M	EX	m s^{-1}	tile	None	meridional
TAUXW	EX	N m^{-2}	tile	None	eastward stress over water
TAUYW	EX	N m^{-2}	tile	None	northward stress over water
OUSTAR3	EX	m^{3} s^{-3}	tile	None	ocean ustar cubed
PENUVF	EX	W m^{-2}	tile	None	downwelling uvr diffuse flux at skin base
PENUVR	EX	W m^{-2}	tile	None	downwelling uvr direct flux at skin base

Short Name	Type	Units	Dims	Vert Loc	Long name
PENPAF	EX	W m^{-2}	tile	None	downwelling par diffuse flux at skin base
PENPAR	EX	W m^{-2}	tile	None	downwelling par direct flux at skin base
TS	IN	K	TileTile	None	surface skin temperature
QS	IN	kg kg^{-1}	TileTile	None	surface specific humidity
FR	IN	1	TileTile	None	ice fraction
CH	IN	kg m^{-2} s^{-1}	TileTile	None	surface heat exchange coefficient
CM	IN	kg m^{-2} s^{-1}	TileTile	None	surface momentum exchange coefficient
CQ	IN	kg m^{-2} s^{-1}	TileTile	None	surface moisture exchange coefficient
ALW	IM	W m^{-2}	tile	None	linearization of surface upwelling longwave flux
BLW	IM	W m^{-2} K^{-1}	tile	None	linearization of surface upwelling longwave flux
DRPAR	IM	W m^{-2}	tile	None	surface downwelling par beam flux
DFPAR	IM	W m^{-2}	tile	None	surface downwelling par diffuse flux
DRNIR	IM	W m^{-2}	tile	None	surface downwelling nir beam flux
DFNIR	IM	W m^{-2}	tile	None	surface downwelling nir diffuse flux
DRUVR	IM	W m^{-2}	tile	None	surface downwelling uvr beam flux
DFUVR	IM	W m^{-2}	tile	None	surface downwelling uvr diffuse flux
LWDNSRF	IM	W m^{-2}	tile	None	surface downwelling longwave flux
EVAP	IM	kg m^{-2} s^{-1}	tile	None	evaporation
SH	IM	W m^{-2}	tile	None	upward sensible heat flux
DEVAP	IM	kg m^{-2} s^{-1}	tile	None	derivative of evaporation
DSH	IM	W m^{-2}	tile	None	derivative of upward sensible heat flux
SNO	IM	kg m^{-2} s^{-1}	tile	None	snowfall
PCU	IM	kg m^{-2} s^{-1}	tile	None	liquid water convective precipitation
PLS	IM	kg m^{-2} s^{-1}	tile	None	liquid water large scale precipitation
TA	IM	K	tile	None	surface air temperature
QA	IM	kg kg^{-1}	tile	None	surface air specific humidity

Short Name	Type	Units	Dims	Vert Loc	Long name
UU	IM	m s^{-1}	tile	None	surface wind speed
UWINDLMTILE	IM	m s^{-1}	tile	None	levellm uwind
VWINDLMTILE	IM	m s^{-1}	tile	None	levellm vwind
DZ	IM	m	tile	None	surface layer height
PS	IM	Pa	tile	None	surface pressure
THATM	IM	K	tile	None	effective surface skin temperature
QHATM	IM	kg kg^{-1}	tile	None	effective surface specific humidity
CTATM	IM	kg m^{-2} s^{-1}	tile	None	surface exchange coefficient for heat
CQATM	IM	kg m^{-2} s^{-1}	tile	None	surface exchange coefficient for moisture

22.2 RUN1 – First Run stage for the Lake component

INTERFACE:

```
subroutine RUN1 ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
  type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
  type(ESMF_State),    intent(inout) :: IMPORT ! Import state
  type(ESMF_State),    intent(inout) :: EXPORT ! Export state
  type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
  integer, optional,   intent(  out) :: RC     ! Error code:
```

DESCRIPTION:
Periodically refreshes the ozone mixing ratios.

22.3 RUN2 – Second Run stage for the Lake component

INTERFACE:

```
subroutine RUN2 ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code:
```

DESCRIPTION:
Periodically refreshes the ozone mixing ratios.

23 Module GEOS_LandGridCompMod – A Module to combine VegDyn and Catch Gridded Components

DESCRIPTION:
This gridded component operates on the land tiles as as child of GEOS_SurfaceGridComp. The core functionality is the calculation of energy and water fluxes impacting the lowest layer of the atmospheric grid. In order to operate on the tilespace specified by its parent, GEOS_LandGridComp runs its child VegdynGridComp to determine relevant time-dependent land-surface characteristics. All parameters calculated in VegdynGridComp are required by CatchGridComp. Furthermore, several exports of the Vegdyn routines are also exports from the Land composite, for use in other modules, such as the case for lai and grn needed in radiation. Vegdyn will be updated first. Then the catchment call will be issued. The composite exports consist of the union of the catchment exports with a subset of the vegdyn exports. All imports and exports are on the prescribed tile grid in the (IM, JM) = (NTILES, 1) convention.
USES:

```
use ESMF
use MAPL_Mod

use GEOS_VegdynGridCompMod,  only : VegdynSetServices    = > SetServices
use GEOS_CatchGridCompMod,   only : CatchSetServices     = > SetServices
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

23.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer, optional                  :: RC  ! return code
```

DESCRIPTION:
The SetServices for the Physics GC needs to register its Initialize and Run. It uses the MAPL_Generic construct for defining state specs and couplings among its children. In addition, it creates the children GCs (VegDyn, Catch) and runs their respective SetServices.
STATES:

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				

Short Name	Type	Units	Dims	Vert Loc	Long name
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				

Short Name	Type	Units	Dims	Vert Loc	Long name
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				
	EX				

23.2 Run1 – First Run method for the composite Land Gridded Component

INTERFACE:

```
subroutine Run1(GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code
```

DESCRIPTION:
This first run method calls the children's first run methods. VEGDYN has only one, and it is called here.

23.3 Run2 – Second Run method for the composite Land Gridded Component

INTERFACE:

```
subroutine Run2(GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code
```

DESCRIPTION:
This second run method call only the catchments second method.

24 Module GEOS_LandiceGridCompMod – Implements slab landice tiles.

! = An improved version over the slab landice
TODO :
- Add multiple elevation classes support to account for ice sheet topo changes
- Add more layers for a more realistic treatment of ice energy budget
- Compute surface mass balance (SMB) as an export for coupling to a dynamic ice-sheet model in future ! = INTERFACE:

USES:

```
use sfclayer   !  use module that contains surface layer routines
use StieglitzSnow,             only: snowrt, SNOW_ALBEDO, TRID
use ESMF
use MAPL_Mod
use GEOS_UtilsMod
use DragCoefficientsMod


integer, parameter :: ICE    =  1
integer, parameter :: SNOW   =  2
integer, parameter :: NUM_SUBTILES  =  2
integer, parameter :: NUM_SNOW_LAYERS  =  15
integer, parameter :: NUM_ICE_LAYERS   =  15
integer, parameter :: NUM_SNOICE_LAYERS  =  NUM_SNOW_LAYERS+NUM_ICE_LAYERS
integer, parameter :: NUM_CONSTIT   =  1
real,    parameter :: rad_to_deg       =  180.0 / 3.1415926


! snowrt related constants
! will move these to a global module later
real,    parameter :: ALHE      =  MAPL_ALHL   ! J/kg  @15C
real,    parameter :: ALHM      =  MAPL_ALHF   ! J/kg
real,    parameter :: TF        =  MAPL_TICE   ! K
real,    parameter :: RHOW      =  MAPL_RHOWTR ! kg/m^3
real,    parameter :: RHOFRESH  =  300.        ! kg/m^3  density of fresh snow
!real,    parameter :: RHOMA     =  500.        ! kg/m^3  maximum snow density
real,    parameter :: RHOICE    =  917.        ! kg/m^3  pure ice density
real,    parameter :: MINSWE    =  0.013       ! kg/m^2  min SWE to avoid immediate melt
real,    parameter :: MAXSNDZ   =  15.0        ! m
real,    parameter :: ZERO      =  0.
real,    parameter :: ONE       =  1.
real,    parameter :: BIG       =  1.e10
real,    parameter :: cpw       =  2065.22    !  @ 0 C [J/kg/K]
real,    parameter :: condice   =  2.25       !  @ 0 C [W/m/K]
real,    parameter :: MINFRACSNO  =  1.e-20   !  mininum sno/ice fraction for
                                            !  heat diffusion of ice layers to take effect
real,    parameter :: LWCTOP      =  1.       !  top thickness to compute LWC. 1m taken from
                                            !  Fettweis et al 2011
! taken from CICE
 real,    parameter :: &                       ! currently used only
        AWTVDR  =  0.00318, &! visible, direct  ! for history and
        AWTIDR  =  0.00182, &! near IR, direct  ! diagnostics
```

```
        AWTVDF  =  0.63282, &! visible, diffuse
        AWTIDF  =  0.36218   ! near IR, diffuse


!real,    dimension(NUM_SNOW_LAYERS), parameter   :: DZMAX  =  (/0.08, 0.12, big/)
real,    dimension(NUM_SNOW_LAYERS), parameter   :: DZMAX  =  (/0.08, 0.08, 0.08 &
           , 0.15, 0.25, big, big, big, big, big, big, big, big, big, big/)
real,    dimension(NUM_ICE_LAYERS), parameter   :: DZMAXI  =  (/0.08, 0.08, 0.08 &
           , 0.15, 0.25, 0.5, 0.75, 1.0, 1.25, 1.5, 1.75, 2.0, 2.5, 3.0, 4.0/)



integer,    parameter :: TAR_PE      =  43
integer,    parameter :: TAR_TILE    =  1
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

DESCRIPTION:
`GEOS_Landice` is a light-weight gridded component that updates the landice tiles

---

24.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
  type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
  integer, optional                  :: RC  ! return code
```

DESCRIPTION:
This version uses the MAPL_GenericSetServices, which sets the Initialize and Finalize services, as well as allocating our instance of a generic state and putting it in the gridded component (GC). Here we only need to set the run method and add the state variable specifications (also generic) to our instance of the generic state. This is the way our true state variables get into the ESMF_State INTERNAL, which is in the MAPL_MetaComp. The import and internal variables are allocated and initialized by generic. Here generic is used for tiles. *STATES:*

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
EMIS	EX	1	tile	None	surface emissivity
ALBVR	EX	1	tile	None	surface albedo for visible beam
ALBVF	EX	1	tile	None	surface albedo for visible diffuse
ALBNR	EX	1	tile	None	surface albedo for near infrared beam
ALBNF	EX	1	tile	None	surface albedo for near infrared diffuse
TST	EX	K	tile	None	surface skin temperature
LST	EX	K	tile	None	land surface skin temperature
QST	EX	kg kg^{-1}	tile	None	surface specific humidity
TH	EX	K	tile	None	turbulence surface skin temperature
QH	EX	kg kg^{-1}	tile	None	turbulence surface specific humidity
DELTS	EX	K	tile	None	change of surface skin temperature
DELQS	EX	kg kg^{-1}	tile	None	change of surface specific humidity
CHT	EX	kg m^{-2} s^{-1}	tile	None	surface heat exchange coefficient
CMT	EX	kg m^{-2} s^{-1}	tile	None	surface momentum exchange coefficient
CQT	EX	kg m^{-2} s^{-1}	tile	None	surface moisture exchange coefficient
CNT	EX	1	tile	None	neutral drag coefficient
RIT	EX	1	tile	None	surface bulk richardson number
ACCUM	EX	kg m^{-2} s^{-1}	tile	None	net ice accumulation rate
EVPICE_GL	EX	W m^{-2}	tile	None	snow ice evaporation energy flux over glaciated surface
SUBLIM	EX	kg m^{-2} s^{-1}	tile	None	sublimation
SNOMAS_GL	EX	kg m^{-2}	tile	None	snow mass over glaciated surface
SNOWMASS	EX	kg m^{-2}	tile	None	snow mass over glaciated surface
SNOWDP_GL	EX	m	tile	None	snow depth over glaciated surface
ASNOW_GL	EX	1	tile	None	fractional area of glaciated surface snowcover
RHOSNOW	EX	kg m^{-3}	tile	None	snow layer density
TSNOW	EX	deg C	tile	None	snow layer temperature

Short Name	Type	Units	Dims	Vert Loc	Long name
TICE0	EX	deg C	tile	None	aggregated ice layer temperature
WSNOW	EX	kg m^{-2}	tile	None	snow layer water content
ZSNOW	EX	m	tile	None	snow layer thickness
DRHOS0	EX	kg m^{-3}	tile	None	snow layer density change due to densification
WESNEX	EX	kg m^{-2} s^{-1}	tile	None	snow layer mass residual due to densification
WESNEXT	EX	kg m^{-2} s^{-1}	tile	None	total snow mass residual due to densification
WESNSC	EX	kg m^{-2} s^{-1}	tile	None	top snow layer mass change due to sub con
SNDZSC	EX	m s^{-1}	tile	None	top snow layer thickness change due to sub con
WESNPREC	EX	kg m^{-2} s^{-1}	tile	None	top snow layer mass change due to precip
SNDZPREC	EX	m s^{-1}	tile	None	top snow layer thickness change due to precip
SNDZ1PERC	EX	m s^{-1}	tile	None	top snow layer thickness change due to percolation
WESNPERC	EX	kg m^{-2} s^{-1}	tile	None	snow layer mass change due to percolation
WESNDENS	EX	kg m^{-2} s^{-1}	tile	None	snow layer mass change due to densification
WESNREPAR	EX	kg m^{-2} s^{-1}	tile	None	snow layer mass change due to repartition
WESNBOT	EX	kg m^{-2} s^{-1}	tile	None	frozen runoff due to fixed max depth
RAINRFZ	EX	kg m^{-2} s^{-1}	tile	None	contribution to smb from refreezed rain over bare ice
SMELT	EX	kg m^{-2} s^{-1}	tile	None	snowmelt flux
IMELT	EX	kg m^{-2} s^{-1}	tile	None	icemelt flux
SNOWALB	EX	1	tile	None	snow broadband albedo
SNICEALB	EX	1	tile	None	aggregated snow ice broadband albedo
MELTWTR	EX	kg m^{-2} s^{-1}	tile	None	melt water production
MELTWTRCONT	EX	kg m^{-2}	tile	None	melt water content
LWC	EX	1	tile	None	liquid water content in top x m
RUNOFF	EX	kg m^{-2} s^{-1}	tile	None	runoff flux
GUST	EX	m s^{-1}	tile	None	gustiness
VENT	EX	m s^{-1}	tile	None	surface ventilation velocity
Z0	EX	m	tile	None	surface roughness

Short Name	Type	Units	Dims	Vert Loc	Long name
Z0H	EX	m	tile	None	surface roughness for heat
MOT2M	EX	K	tile	None	temperature
MOQ2M	EX	kg kg^{-1}	tile	None	humidity
MOU2M	EX	m s^{-1}	tile	None	zonal
MOV2M	EX	m s^{-1}	tile	None	meridional
MOT10M	EX	K	tile	None	temperature
MOQ10M	EX	kg kg^{-1}	tile	None	humidity
MOU10M	EX	m s^{-1}	tile	None	zonal
MOV10M	EX	m s^{-1}	tile	None	meridional
MOU50M	EX	m s^{-1}	tile	None	zonal
MOV50M	EX	m s^{-1}	tile	None	meridional
EVAPOUT	EX	kg m^{-2} s^{-1}	tile	None	evaporation
SHOUT	EX	W m^{-2}	tile	None	upward sensible heat flux
HLWUP	EX	W m^{-2}	tile	None	surface outgoing longwave flux
LWNDSRF	EX	W m^{-2}	tile	None	surface net downward longwave flux
SWNDSRF	EX	W m^{-2}	tile	None	surface net downward shortwave flux
HLATN	EX	W m^{-2}	tile	None	total latent energy flux
DNICFLX	EX	W m^{-2}	tile	None	downward heat flux in ice
GHSNOW	EX	W m^{-2}	tile	None	Ground heating snow
GHTSKIN	EX	W m^{-2}	tile	None	Ground heating for tskin
ITY	EX	1	tile	None	vegetation type
TS	IN	K	tile	None	surface skin temperature
QS	IN	kg kg^{-1}	tile	None	surface specific humidity
FR	IN	1	tile	None	ice fraction
CH	IN	kg m^{-2} s^{-1}	tile	None	surface heat exchange coefficient
CM	IN	kg m^{-2} s^{-1}	tile	None	surface momentum exchange coefficient
CQ	IN	kg m^{-2} s^{-1}	tile	None	surface moisture exchange coefficient
WESN	IN	kg m^{-2}	tile	None	snow layer mass
HTSN	IN	J m^{-2}	tile	None	snow layer heat content
SNDZ	IN	m	tile	None	snow layer depth
TICE	IN	k	tile	None	ice layer temperature
ALW	IM	W m^{-2}	tile	None	linearization of surface upwelling longwave flux
BLW	IM	W m^{-2} K^{-1}	tile	None	linearization of surface upwelling longwave flux
DRPAR	IM	W m^{-2}	tile	None	surface downwelling par beam flux

Short Name	Type	Units	Dims	Vert Loc	Long name
DFPAR	IM	W m^{-2}	tile	None	surface downwelling par diffuse flux
DRNIR	IM	W m^{-2}	tile	None	surface downwelling nir beam flux
DFNIR	IM	W m^{-2}	tile	None	surface downwelling nir diffuse flux
DRUVR	IM	W m^{-2}	tile	None	surface downwelling uvr beam flux
DFUVR	IM	W m^{-2}	tile	None	surface downwelling uvr diffuse flux
LWDNSRF	IM	W m^{-2}	tile	None	surface downwelling longwave flux
EVAP	IM	kg m^{-2} s^{-1}	tile	None	evaporation
SH	IM	W m^{-2}	tile	None	upward sensible heat flux
DEVAP	IM	kg m^{-2} s^{-1}	tile	None	derivative of evaporation
DSH	IM	W m^{-2}	tile	None	derivative of upward sensible heat flux
SNO	IM	kg m^{-2} s^{-1}	tile	None	snowfall
TA	IM	K	tile	None	surface air temperature
QA	IM	kg kg^{-1}	tile	None	surface air specific humidity
UU	IM	m s^{-1}	tile	None	surface wind speed
UWINDLMTILE	IM	m s^{-1}	tile	None	levellm uwind
VWINDLMTILE	IM	m s^{-1}	tile	None	levellm vwind
DZ	IM	m	tile	None	surface layer height
PS	IM	Pa	tile	None	surface pressure
PCU	IM	kg m^{-2} s^{-1}	tile	None	liquid water convective precipitation
PLS	IM	kg m^{-2} s^{-1}	tile	None	liquid water large scale precipitation
THATM	IM	K	tile	None	effective surface skin temperature
QHATM	IM	kg kg^{-1}	tile	None	effective surface specific humidity
CTATM	IM	kg m^{-2} s^{-1}	tile	None	surface exchange coefficient for heat
CQATM	IM	kg m^{-2} s^{-1}	tile	None	surface exchange coefficient for moisture

24.2 RUN1 – First Run stage for the LandIce component

INTERFACE:

```
subroutine RUN1 ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code:
```

DESCRIPTION:
Periodically refreshes the ozone mixing ratios.

24.3 RUN2 – Second Run method for the LandIce component

INTERFACE:

```
subroutine RUN2 ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code:
```

DESCRIPTION:
Periodically refreshes the ozone mixing ratios.

25 Module GEOS_Moist – A Module to compute moist processes, including convection,

large-scale condensation and precipitation and cloud parameters. INTERFACE:

```
module GEOSMoistGridCompMod
```

USES:

```
use RAS       ! using module that contains ras code

use CLOUDNEW, only: PROGNO_CLOUD, ICE_FRACTION, T_CLOUD_CTL
use CLOUDNEW, only: &
```

```
! Subroutines
PROGNO_CLOUD, ICE_FRACTION, &
! Derived Data Types
T_CLOUD_CTL, &
! Inputs
PP_DEV, EXNP_DEV, PPE_DEV, KH_DEV, FRLAND_DEV, &
RMFDTR_DEV, QLWDTR_DEV, U_DEV, V_DEV, QST3_DEV, &
DZET_DEV, QDDF3_DEV, TEMPOR_DEV, &
! Inoutputs
TH_DEV, Q_DEV, QRN_CU_DEV, CNV_UPDFRC_DEV, QLW_LS_DEV, &
QLW_AN_DEV, QIW_LS_DEV, QIW_AN_DEV, ANVFRC_DEV, CLDFRC_DEV, &
! Outputs
RAD_CLDFRC_DEV, RAD_QL_DEV, RAD_QI_DEV, RAD_QR_DEV, RAD_QS_DEV, &
CLDREFFL_DEV, CLDREFFI_DEV, PRELS_DEV, PRECU_DEV, PREAN_DEV, &
LSARF_DEV, CUARF_DEV, ANARF_DEV, SNRLS_DEV, SNRCU_DEV, SNRAN_DEV, &
! Working arrays
PFL_CN_DEV, PFI_CN_DEV, PFL_AN_DEV, &
PFI_AN_DEV, PFL_LS_DEV, PFI_LS_DEV, &
! Diagnostics
RHX_DEV, &
REV_LS_DEV, REV_AN_DEV, REV_CN_DEV, &
RSU_LS_DEV, RSU_AN_DEV, RSU_CN_DEV, &
ACLL_CN_DEV, ACIL_CN_DEV, ACLL_AN_DEV, &
ACIL_AN_DEV, ACLL_LS_DEV, ACIL_LS_DEV, &
PDFL_DEV, PDFI_DEV, FIXL_DEV, FIXI_DEV, &
AUT_DEV, EVAPC_DEV, SDM_DEV, &
SUBLC_DEV, FRZ_TT_DEV, DCNVL_DEV, DCNVI_DEV, &
ALPHT_DEV, ALPH1_DEV, ALPH2_DEV, &
CFPDF_DEV, RHCLR_DEV, DQRL_DEV, FRZ_PP_DEV, &
VFALLICE_AN_DEV, VFALLICE_LS_DEV, &
VFALLWAT_AN_DEV, VFALLWAT_LS_DEV, &
VFALLSN_AN_DEV, VFALLSN_LS_DEV, &
VFALLSN_CN_DEV, VFALLRN_AN_DEV, &
VFALLRN_LS_DEV, VFALLRN_CN_DEV, &
!CFPDFX is no longer calculated in PROGNO_CLOUD, but still an export
!CFPDFX_DEV, &

! Constants
! PHYSPARAMS Constants are loaded into constant memory
CNV_BETA, ANV_BETA, LS_BETA, RH00, C_00, LWCRIT, C_ACC, &
C_EV_R, C_EV_S, CLDVOL2FRC, RHSUP_ICE, SHR_EVAP_FAC, MIN_CLD_WATER, &
CLD_EVP_EFF, NSMAX, LS_SDQV2, LS_SDQV3, LS_SDQVT1, ANV_SDQV2, &
ANV_SDQV3, ANV_SDQVT1, ANV_TO_LS, N_WARM, N_ICE, N_ANVIL, &
N_PBL, DISABLE_RAD, ANV_ICEFALL_C, LS_ICEFALL_C, REVAP_OFF_P, CNVENVFC, &
WRHODEP, T_ICE_ALL, CNVICEPARAM, ICEFRPWR, CNVDDRFC, ANVDDRFC, &
```

```
        LSDDRFC, TANHRHCRIT, MINRHCRIT, MAXRHCRIT, TURNRHCRIT, MAXRHCRITLAND, &
        FR_LS_WAT, FR_LS_ICE, FR_AN_WAT, FR_AN_ICE, MIN_RL, MIN_RI, MAX_RL, &
        MAX_RI, RI_ANV, PDFFLAG
  use cudafor

  use DDF

  use ESMF
  use MAPL_Mod
  use GEOS_UtilsMod
```

PUBLIC MEMBER FUNCTIONS:

```
  public SetServices
```

DESCRIPTION:
GEOS_MoistGridCompMod implements moist processes in GEOS-5. These include all processes that involve phase changes in the atmosphere, such as large-scale condensation, convective clouds, and all rain and cloud formation. It's state consists of water vapor, various types of condensate, and fractions of various cloud types.

25.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
  subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
    type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
    integer, optional                  :: RC  ! return code
```

DESCRIPTION:
GEOS_MoistGridCompMod uses the default Initialize and Finalize services, but registers its own Run method. *STATES:*

The following is a list of Import, Export and Internal states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
Q	IN	kg kg^{-1}	xyz	Center	specific humidity

Short Name	Type	Units	Dims	Vert Loc	Long name
QLLS	IN	kg kg^{-1}	xyz	Center	mass fraction of large scale cloud liquid water
QLCN	IN	kg kg^{-1}	xyz	Center	mass fraction of convective cloud liquid water
CLLS	IN	1	xyz	Center	large scale cloud area fraction
CLCN	IN	1	xyz	Center	convective cloud area fraction
QILS	IN	kg kg^{-1}	xyz	Center	mass fraction of large scale cloud ice water
QICN	IN	kg kg^{-1}	xyz	Center	mass fraction of convective cloud ice water
$\overline{\text{PLE}}$	IM	Pa	xyz	Edge	air pressure
$\overline{\text{PREF}}$	IM	Pa	z	Edge	reference air pressure
$\overline{\text{KH}}$	IM	m^2 s^{-1}	xyz	Edge	scalar diffusivity
$\overline{\text{TH}}$	IM	K	xyz	Center	potential temperature
$\overline{\text{U}}$	IM	m s^{-1}	xyz	Center	eastward wind
$\overline{\text{V}}$	IM	m s^{-1}	xyz	Center	northward wind
$\overline{\text{TS}}$	IM	K	xy	None	surface
$\overline{\text{FRLAND}}$	IM	1	xy	None	areal land fraction
$\overline{\text{FROCEAN}}$	IM	1	xy	None	areal ocean fraction
MTR	IM	X	xyz	Center	tracers for moist
$\overline{\text{KPBL}}$	IM	1	xy	None	planetary boundary layer level
QCTOT	EX	kg kg^{-1}	xyz	Center	mass fraction of total cloud water
QLTOT	EX	kg kg^{-1}	xyz	Center	grid box mass fraction of cloud liquid water
QITOT	EX	kg kg^{-1}	xyz	Center	grid box mass fraction of cloud ice water
QRTOT	EX	kg kg^{-1}	xyz	Center	mass fraction of falling rain
QSTOT	EX	kg kg^{-1}	xyz	Center	mass fraction of falling snow
MTRI	EX	X s^{-1}	xyz	Center	tracer tendencies due to moist
DTHDT	EX	Pa K s^{-1}	xyz	Center	pressure weighted potential temperature tendency due to moist
DTDTFRIC	EX	Pa K s^{-1}	xyz	Center	pressure weighted temperature tendency due to moist friction
DQDT	EX	kg kg^{-1} s^{-1}	xyz	Center	specific humidity tendency due to moist
DUDT	EX	m s^{-2}	xyz	Center	zonal wind tendency due to moist
DVDT	EX	m s^{-2}	xyz	Center	meridional wind tendency due to moist

Short Name	Type	Units	Dims	Vert Loc	Long name
DTHDTCN	EX	K s^{-1}	xyz	Center	potential temperature tendency due to convection
DQDTCN	EX	kg kg^{-1} s^{-1}	xyz	Center	specific humidity tendency due to convection
DQLDT	EX	kg kg^{-1} s^{-1}	xyz	Center	total liq water tendency due to moist
DQIDT	EX	kg kg^{-1} s^{-1}	xyz	Center	total ice water tendency due to moist
DQCDTCN	EX	kg kg^{-1} s^{-1}	xyz	Center	condensate tendency due to convection
CNV_DQLDT	EX	kg m^{-2} s^{-1}	xyz	Center	convective condensate source
CNV_PRC3	EX	kg m^{-2} s^{-1}	xyz	Center	convective precipitation from RAS
DQRL	EX	kg kg^{-1} s^{-1}	xyz	Center	large scale rainwater source
DQRC	EX	kg kg^{-1} s^{-1}	xyz	Center	convective rainwater source
CNV_MF0	EX	kg m^{-2} s^{-1}	xyz	Center	cloud base mass flux
CNV_MFD	EX	kg m^{-2} s^{-1}	xyz	Center	detraining mass flux
CNV_MFC	EX	kg m^{-2} s^{-1}	xyz	Edge	cumulative mass flux
CNV_FREQ	EX	fraction	xy	None	convective frequency
CNV_BASEP	EX	Pa	xy	None	pressure at convective cloud base
CNV_TOPP	EX	Pa	xy	None	pressure at convective cloud top
CNV_UPDF	EX	1	xyz	Center	updraft areal fraction
CNV_CVW	EX	hPa s^{-1}	xyz	Center	updraft vertical velocity
CNV_QC	EX	kg kg^{-1}	xyz	Center	grid mean convective condensate
RL	EX	m	xyz	Center	liquid cloud particle effective radius
RI	EX	m	xyz	Center	ice phase cloud particle effective radius
RR	EX	m	xyz	Center	falling rain particle effective radius
RS	EX	m	xyz	Center	falling ice particle effective radius
CLDNCCN	EX	m^{-3}	xyz	Center	number concentration of cloud particles
QSATI	EX	kg kg^{-1}	xyz	Center	saturation spec hum over ice
QSATL	EX	kg kg^{-1}	xyz	Center	saturation spec hum over liquid

Short Name	Type	Units	Dims	Vert Loc	Long name
ALPHT	EX	1	xyz	Center	pdf spread for condensation over qsat total
ALPH1	EX	1	xyz	Center	pdf spread for condensation over qsat term1
ALPH2	EX	1	xyz	Center	pdf spread for condensation over qsat term2
CFPDFX	EX	1	xyz	Center	cloud fraction internal in PDF scheme
RHCLR	EX	1	xyz	Center	RH clear sky
CFPDF	EX	1	xyz	Center	cloud fraction after PDF
FCLD	EX	1	xyz	Center	cloud fraction for radiation
QV	EX	kg kg^{-1}	xyz	Center	water vapor for radiation
QL	EX	kg kg^{-1}	xyz	Center	in cloud cloud liquid for radiation
QI	EX	kg kg^{-1}	xyz	Center	in cloud cloud ice for radiation
QR	EX	kg kg^{-1}	xyz	Center	Falling rain for radiation
QS	EX	kg kg^{-1}	xyz	Center	Falling snow for radiation
LS_PRCP	EX	kg m^{-2} s^{-1}	xy	None	nonanvil large scale precipitation
AN_PRCP	EX	kg m^{-2} s^{-1}	xy	None	anvil precipitation
CN_PRCP	EX	kg m^{-2} s^{-1}	xy	None	convective precipitation
ER_PRCP	EX	kg m^{-2} s^{-1}	xy	None	spurious rain from RH cleanup
FILLNQV	EX	kg m^{-2} s^{-1}	xy	None	filling of negative Q
PGENTOT	EX	kg m^{-2} s^{-1}	xy	None	Total column production of precipitation
PREVTOT	EX	kg m^{-2} s^{-1}	xy	None	Total column re-evap/subl of precipitation
LS_ARF	EX	1	xy	None	areal fraction of nonanvil large scale showers
AN_ARF	EX	1	xy	None	areal fraction of anvil showers
CN_ARF	EX	1	xy	None	areal fraction of convective showers
SNO	EX	kg m^{-2} s^{-1}	xy	None	snowfall
PCU	EX	kg m^{-2} s^{-1}	xy	None	convective rainfall
PLS	EX	kg m^{-2} s^{-1}	xy	None	large scale rainfall
TPREC	EX	kg m^{-2} s^{-1}	xy	None	total precipitation
HOURNORAIN	EX	s	xy	None	time-during an hour with no precipitation
TPW	EX	kg m^{-2}	xy	None	total precipitable water
CCWP	EX	kg m^{-2}	xy	None	grid mean conv cond water path diagnostic

Short Name	Type	Units	Dims	Vert Loc	Long name
CWP	EX	kg m^{-2}	xy	None	condensed water path
LWP	EX	kg m^{-2}	xy	None	liquid water path
IWP	EX	kg m^{-2}	xy	None	ice water path
BYNCY	EX	m s^{-2}	xyz	Center	buoyancy of
CAPE	EX	J m^{-2}	xy	None	cape for surface parcel
INHB	EX	1	xy	None	inhibition for surface parcel
TVQ0	EX	kg m^{-2}	xy	None	Total Water Substance Before
TVQ1	EX	kg m^{-2}	xy	None	Total Water Substance After
DCPTE	EX	J m^{-2}	xy	None	Total VI DcpT
TVE0	EX	J m^{-2}	xy	None	Total VI MSE Before
TVE1	EX	J m^{-2}	xy	None	Total VI MSE After
TVEX	EX	J m^{-2}	xy	None	Total VI MSE Somewhere
ZPBLCN	EX	m	xy	None	boundary layer depth
ZLCL	EX	m	xy	None	lifting condensation level
ZLFC	EX	m	xy	None	level of free convection
ZCBL	EX	m	xy	None	height of cloud base layer
MXDIAM	EX	m	xy	None	diameter of largest RAS plume
RASTIME	EX	s	xy	None	timescale for deep RAS plumes
RASPBLQ	EX	(m^3 s^{-1})/2	xy	None	sqrt of integral KH dz
ENTLAM	EX	kg m^{-2} s^{-1}	xyz	Center	entrainment
QLCN	EX	1	xyz	Center	mass fraction of convective cloud liquid water
QICN	EX	1	xyz	Center	mass fraction of convective cloud ice water
CLLS	EX	1	xyz	Center	large scale cloud area fraction
CLCN	EX	1	xyz	Center	convective cloud area fraction
RH1	EX	1	xyz	Center	relative humidity before moist
RH2	EX	1	xyz	Center	relative humidity after moist
QILS_tlad	EX	kg kg^{-1}	xyz	Center	total cloud liquid ice before moist
QLLS_tlad	EX	kg kg^{-1}	xyz	Center	total anvil liquid ice before moist
QICN_tlad	EX	kg kg^{-1}	xyz	Center	total cloud liquid water before moist
QLCN_tlad	EX	kg kg^{-1}	xyz	Center	total anvil liquid water before moist
CFLS_tlad	EX	1	xyz	Center	large scale cloud fraction before moist
CFCN_tlad	EX	1	xyz	Center	convective cloud fraction before moist
KCBL_tlad	EX	1	xy	None	KCBL before moist
TSUR_tlad	EX	K	xy	None	surface temp before moist

Short Name	Type	Units	Dims	Vert Loc	Long name
QLS_tlad	EX	kg kg^{-1}	xyz	Center	total cloud liquid ice before moist
QCN_tlad	EX	kg kg^{-1}	xyz	Center	total cloud liquid water before moist
RHX	EX	1	xyz	Center	relative humidity after PDF
REVSU_CN	EX	kg kg^{-1} s^{-1}	xyz	Center	evap subl of convective precipitation
REVSU_LSAN	EX	kg kg^{-1} s^{-1}	xyz	Center	evap subl of non convective precipitation
REV_CN	EX	kg kg^{-1} s^{-1}	xyz	Center	evaporation of convective precipitation
REV_AN	EX	kg kg^{-1} s^{-1}	xyz	Center	evaporation of anvil precipitation
REV_LS	EX	kg kg^{-1} s^{-1}	xyz	Center	evaporation of nonanvil large scale precipitation
RSU_CN	EX	kg kg^{-1} s^{-1}	xyz	Center	sublimation of convective precipitation
RSU_AN	EX	kg kg^{-1} s^{-1}	xyz	Center	sublimation of anvil precipitation
RSU_LS	EX	kg kg^{-1} s^{-1}	xyz	Center	sublimation of nonanvil large scale precipitation
ACR_TOT	EX	kg kg^{-1} s^{-1}	xyz	Center	total accretion of precipitation
ACRLL_CN	EX	kg kg^{-1} s^{-1}	xyz	Center	liq liq accretion of convective precipitation
ACRLL_AN	EX	kg kg^{-1} s^{-1}	xyz	Center	liq liq accretion of anvil precipitation
ACRLL_LS	EX	kg kg^{-1} s^{-1}	xyz	Center	liq liq accretion of nonanvil large scale precipitation
ACRIL_CN	EX	kg kg^{-1} s^{-1}	xyz	Center	ice liq accretion of convective precipitation
ACRIL_AN	EX	kg kg^{-1} s^{-1}	xyz	Center	ice liq accretion of anvil precipitation
ACRIL_LS	EX	kg kg^{-1} s^{-1}	xyz	Center	ice liq accretion of nonanvil large scale precipitation
PFI_CN	EX	kg m^{-2} s^{-1}	xyz	Edge	3D flux of ice convective precipitation
PFI_AN	EX	kg m^{-2} s^{-1}	xyz	Edge	3D flux of ice anvil precipitation
PFI_LS	EX	kg m^{-2} s^{-1}	xyz	Edge	3D flux of ice nonanvil large scale precipitation
PFI_LSAN	EX	kg m^{-2} s^{-1}	xyz	Edge	3D flux of ice nonconvective precipitation

Short Name	Type	Units	Dims	Vert Loc	Long name
PFL_CN	EX	kg m^{-2} s^{-1}	xyz	Edge	3D flux of liquid convective precipitation
PFL_AN	EX	kg m^{-2} s^{-1}	xyz	Edge	3D flux of liquid anvil precipitation
PFL_LS	EX	kg m^{-2} s^{-1}	xyz	Edge	3D flux of liquid nonanvil large scale precipitation
PFL_LSAN	EX	kg m^{-2} s^{-1}	xyz	Edge	3D flux of liquid nonconvective precipitation
DCNVL	EX	kg kg^{-1} s^{-1}	xyz	Center	convective source of cloud liq
DCNVI	EX	kg kg^{-1} s^{-1}	xyz	Center	convective source of cloud ice
DLPDF	EX	kg kg^{-1} s^{-1}	xyz	Center	pdf source sink of cloud liq
DIPDF	EX	kg kg^{-1} s^{-1}	xyz	Center	pdf source sink of cloud ice
DLFIX	EX	kg kg^{-1} s^{-1}	xyz	Center	fix source sink of cloud liq
DIFIX	EX	kg kg^{-1} s^{-1}	xyz	Center	fix source sink of cloud ice
AUT	EX	kg kg^{-1} s^{-1}	xyz	Center	autoconv sink of cloud liq
EVAPC	EX	kg kg^{-1} s^{-1}	xyz	Center	evaporation of cloud liq
SDM	EX	kg kg^{-1} s^{-1}	xyz	Center	sedimentation sink of cloud ice
VFALLICE_AN	EX	m s^{-1}	xyz	Center	autoconversion fall velocity of anvil snow
VFALLICE_LS	EX	m s^{-1}	xyz	Center	autoconversion fall velocity of largescale snow
VFALLWAT_AN	EX	m s^{-1}	xyz	Center	autoconversion fall velocity of anvil rain
VFALLWAT_LS	EX	m s^{-1}	xyz	Center	autoconversion fall velocity of largescale rain
VFALLRN_AN	EX	m s^{-1}	xyz	Center	reevaporation fall velocity of anvil rain
VFALLRN_LS	EX	m s^{-1}	xyz	Center	reevaporation fall velocity of largescale rain
VFALLRN_CN	EX	m s^{-1}	xyz	Center	reevaporation fall velocity of convective rain
VFALLSN_AN	EX	m s^{-1}	xyz	Center	reevaporation fall velocity of anvil snow

Short Name	Type	Units	Dims	Vert Loc	Long name
VFALLSN_LS	EX	m s^{-1}	xyz	Center	reevaporation fall velocity of largescale snow
VFALLSN_CN	EX	m s^{-1}	xyz	Center	reevaporation fall velocity of convective snow
SUBLC	EX	kg kg^{-1} s^{-1}	xyz	Center	sublimation of cloud ice
FRZ_TT	EX	kg kg^{-1} s^{-1}	xyz	Center	freezing of cloud condensate
FRZ_PP	EX	kg kg^{-1} s^{-1}	xyz	Center	freezing of precip condensate
PDFLZ	EX	kg m^{-2} s^{-1}	xy	None	statistical source of cloud water
PDFIZ	EX	kg m^{-2} s^{-1}	xy	None	statistical source of cloud ice
CNVRNZ	EX	kg m^{-2} s^{-1}	xy	None	convective production of rain water
CNVLZ	EX	kg m^{-2} s^{-1}	xy	None	convective source of cloud water
CNVIZ	EX	kg m^{-2} s^{-1}	xy	None	convective source of cloud ice
EVPCZ	EX	kg m^{-2} s^{-1}	xy	None	evaporation loss of cloud water
SUBCZ	EX	kg m^{-2} s^{-1}	xy	None	sumblimation loss of cloud ice
EVPPZ	EX	kg m^{-2} s^{-1}	xy	None	evaporation loss of precip water
SUBPZ	EX	kg m^{-2} s^{-1}	xy	None	sumblimation loss of precip ice
AUTZ	EX	kg m^{-2} s^{-1}	xy	None	autoconversion loss of cloud water
SDMZ	EX	kg m^{-2} s^{-1}	xy	None	sedimentation loss of cloud ice
COLLLZ	EX	kg m^{-2} s^{-1}	xy	None	accretion loss of cloud water to rain
COLLIZ	EX	kg m^{-2} s^{-1}	xy	None	accretion loss of cloud water to snow
FRZCZ	EX	kg m^{-2} s^{-1}	xy	None	net freezing of cloud condensate
FRZPZ	EX	kg m^{-2} s^{-1}	xy	None	net freezing of precip condensate
RCCODE	EX	codes	xyz	Center	Convection return codes
TRIEDLV	EX	0 or 1	xyz	Center	Tested for convection at this level
QVRAS	EX	kg kg^{-1}	xyz	Center	water vapor after ras

Short Name	Type	Units	Dims	Vert Loc	Long name
THRAS	EX	K	xyz	Center	potential temperature after ras
URAS	EX	m s^{-1}	xyz	Center	eastward wind after ras
VRAS	EX	m s^{-1}	xyz	Center	northward wind after ras
THOI	EX	K	xyz	Center	potential temperature before ras
QHOI	EX	kg kg^{-1}	xyz	Center	specific humidity before ras
QSSI	EX	kg kg^{-1}	xyz	Center	saturation specific humidity before ras
DQSI	EX	kg kg^{-1} K^{-1}	xyz	Center	deriv sat specific humidity wrt t before ras
PLEI	EX	Pa	xyz	Edge	air pressure before ras
TPERTI	EX	K	xy	None	temperature perturbation before ras
KCBLI	EX	1	xy	None	cloud base layer before ras
QX0	EX	kg kg^{-1}	xyz	Center	specific humidity
QLLSX0	EX	kg kg^{-1}	xyz	Center	initial mass fraction of large scale cloud liquid water
QLLSX1	EX	kg kg^{-1}	xyz	Center	final mass fraction of large scale cloud liquid water
QLCNX0	EX	kg kg^{-1}	xyz	Center	initial mass fraction of convective cloud liquid water
QLCNX1	EX	kg kg^{-1}	xyz	Center	final mass fraction of convective cloud liquid water
CLLSX0	EX	1	xyz	Center	large scale cloud area fraction
CLCNX0	EX	1	xyz	Center	convective cloud area fraction
QILSX0	EX	kg kg^{-1}	xyz	Center	initial mass fraction of large scale cloud ice water
QILSX1	EX	kg kg^{-1}	xyz	Center	final mass fraction of large scale cloud ice water
QICNX0	EX	kg kg^{-1}	xyz	Center	initial mass fraction of convective cloud ice water
QICNX1	EX	kg kg^{-1}	xyz	Center	final mass fraction of convective cloud ice water
KHX0	EX	m^2 s^{-1}	xyz	Edge	scalar diffusivity
THX0	EX	K	xyz	Center	potential temperature
UX0	EX	m s^{-1}	xyz	Center	eastward wind
VX0	EX	m s^{-1}	xyz	Center	northward wind
TSX0	EX	K	xy	None	surface
FRLANDX0	EX	1	xy	None	areal land fraction
DDF_MFC	EX	kg m^{-2} s^{-1}	xyz	Edge	Downdraft mass flux
DDF_RH1	EX	1	xyz	Center	Downdraft in cloud RH before

Short Name	Type	Units	Dims	Vert Loc	Long name
DDF_RH2	EX	1	xyz	Center	Downdraft in cloud RH after
DDF_TC	EX	K	xyz	Center	Temperature excess in DDF
DDF_QVC	EX	kg kg^{-1}	xyz	Center	Spec hum excess in DDF
DDF_BYNC	EX	m s^{-2}	xyz	Center	Buoyancy of DDF
DDF_MUPH	EX	kg kg^{-1} s^{-1}	xyz	Center	Downdraft moistening from evap subl
DDF_DQDT	EX	kg kg^{-1} s^{-1}	xyz	Center	Total Downdraft moistening
DDF_DTDT	EX	K s^{-1}	xyz	Center	Total Downdraft heating
DDF_ZSCALE	EX	m	xy	None	vertical scale for downdraft
KEDISS	EX	W m^{-2}	xyz	Center	kinetic energy diss in RAS
KEMST	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency across moist
KEMST2	EX	W m^{-2}	xy	None	vertically integrated KE dissipation in RAS
DDUDT	EX	kg m^{-2} s^{-1}	xy		dust tendency due to conv scav
DSSDT	EX	kg m^{-2} s^{-1}	xy		sea salt tendency due to conv scav
DOCDT	EX	kg m^{-2} s^{-1}	xy		organic carbon tendency due to conv scav
DBCDT	EX	kg m^{-2} s^{-1}	xy		black carbon tendency due to conv scav
DSUDT	EX	kg m^{-2} s^{-1}	xy		sulfate tendency due to conv scav
DDUDTcarma	EX	kg m^{-2} s^{-1}	xy		carma dust tendency due to conv scav
DSSDTcarma	EX	kg m^{-2} s^{-1}	xy		carma seasalt tendency due to conv scav
LFR	EX	km^{-2} s^{-1}	xy	None	lightning flash rate
A1X1	EX	km^{-2} s^{-1}	xy	None	LFR Term number 1
A2X2	EX	km^{-2} s^{-1}	xy	None	LFR Term number 2
A3X3	EX	km^{-2} s^{-1}	xy	None	LFR Term number 3
A4X4	EX	km^{-2} s^{-1}	xy	None	LFR Term number 4
A5X5	EX	km^{-2} s^{-1}	xy	None	LFR Term number 5

25.2 RUN – Run method for the CONVECT component

INTERFACE:

```
subroutine RUN ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code:
```

DESCRIPTION:
This version uses the MAPL_GenericSetServices. This function sets the Initialize and Finalize services, as well as allocating

---

DESCRIPTION:
Generate lightning flash rates [km^{-2} s^{-1}] using a six-variable polynomial fit.

ORIGIN AND CONTACT
Dr. Dale Allen, Associate Research Scientist
Dept. of Atmospheric and Oceanic Science
University of Maryland
College Park, MD 20742
301-405-7629 (ph); 301-314-9482 (fax)
http://www.meto.umd.edu/ allen

FORMULATION NOTES
Predictor variables are set to zero where CN_PRCP is zero or where the optical depth cloud top height is less than 5.5 km. The fit returns flash rates in units km^{-2} day^{-1}. Convert to km^{-2} s^{-1} for the export state.

OTHER NOTES OF INTEREST
MOIST sets CNV_TOPP to zero if there is an absence of convection.

26 Module GEOS_OceanbiogeochemGridCompMod – Implements ocean biology

INTERFACE:

```
module GEOSOceanbiogeochemGridCompMod
```

USES:

```
use ESMF
use MAPL_Mod
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

DESCRIPTION:
GEOS_Obio is a light-weight gridded component does ocean biology

26.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer, optional                  :: RC  ! return code
```

DESCRIPTION:
This version uses the MAPL_GenericSetServices, which sets the Initialize and Finalize services, as well as allocating our instance of a generic state and putting it in the gridded component (GC). Here we only need to set the run method and add the state variable specifications (also generic) to our instance of the generic state. This is the way our true state variables get into the ESMF_State INTERNAL, which is in the MAPL_MetaComp. The import and internal variables are allocated and initialized by generic. Here generic is used for tiles. *STATES:*

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
TURB	EX	m^{-1}	xy	None	water turbidity

26.2 RUN – First Run stage for the Obio component

INTERFACE:

```
subroutine RUN ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
```

```
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code:
```

DESCRIPTION:
Periodically refreshes the penetratin radiation

27 Module GEOS_Ogcm – A composite component for the ogcm components.

INTERFACE:

```
module GEOSOgcmGridCompMod
```

USES:

```
use ESMF
use MAPL_Mod

use OBIO_MODULE,                      only : ObioSetServices    = > SetServices
use ORAD_MODULE,                      only : OradSetServices    = > SetServices

use OCEAN_MODULE,                     only : OceanSetServices   = > SetServices
use SEAICE_MODULE,                    only : SeaIceSetServices  = > SetServices
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

DESCRIPTION:
`GEOS_Ogcm` is a light-weight gridded component that implements the interface to the ogcm components. The ogcm computational components (Poseidon, OceanRadiation, OceanBio-Geochemistry, etc) are its children. This component currently serves as an interface between the exchange grid and the ocean's grid. Its "natural" grid is the ocean part of the exchange grid, and all its imports and exports are on this grid. The natural grid of all of its children is currently the ocean's rectangular grid. The ESMF grid that is in the gridded component is created by the parent and it is the ocean's rectangular grid. At present the exchange grid information is kept in the generic state.
The fact that some of these are friendlies—all the "skin" components—means that it cannot be a "no-work-no-change" component. The interpolation of these to the ocean grid leave an ocean grid imprint on them. No such happens on the atmospheric side. So we should think of these exchange grid friendlies as ocean variables.

27.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer, optional                  :: RC  ! return code
```

DESCRIPTION:
This version uses the MAPL_GenericSetServices, which in addition to setting default IRF methods, also allocates our instance of a generic state and puts it in the gridded component (GC). Here we override the Initialize and Run methods. *STATES:*

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
TAUXW	IM	N m^{-2}	tile	None	eastward stress on ocean
TAUYW	IM	N m^{-2}	tile	None	northward stress on ocean
TAUXI	IM	N m^{-2}	tile	None	eastward stress on ice
TAUYI	IM	N m^{-2}	tile	None	northward stress on ice
OUSTAR3	IM	m^3 s^{-3}	tile	None	ocean ustar cubed
UU	IM	m s^{-1}	tile	None	surface wind speed
PS	IM	Pa	tile	None	surface air pressure
PENUVR	IM	W m^{-2}	tile	None	net downward penetrating direct UV flux
PENPAR	IM	W m^{-2}	tile	None	net downward penetrating direct PAR flux
PENUVF	IM	W m^{-2}	tile	None	net downward penetrating diffuse UV flux
PENPAF	IM	W m^{-2}	tile	None	net downward penetrating diffuse PAR flux
DISCHRG	IM	kg m^{-2} s^{-1}	tile	None	river discharge at ocean points
CO2SC	IM	$1e^{-6}$	tile	None	CO2
DUDP	IM	kg m^{-2} s^{-1}	tile	None	Dust
DUWT	IM	kg m^{-2} s^{-1}	tile	None	Dust
DUSD	IM	kg m^{-2} s^{-1}	tile	None	Dust
BCDP	IM	kg m^{-2} s^{-1}	tile	None	Black
BCWT	IM	kg m^{-2} s^{-1}	tile	None	Black
OCDP	IM	kg m^{-2} s^{-1}	tile	None	Organic

Short Name	Type	Units	Dims	Vert Loc	Long name
OCWT	IM	kg m^{-2} s^{-1}	tile	None	Organic
FSWBAND	IM	W m^{-2}	tile	None	net surface downward shortwave flux per band in air
FSWBANDNA	IM	W m^{-2}	tile	None	net surface downward shortwave flux per band in air assuming no aerosol
HW	IM	kg	tile	None	water skin layer mass
TW	IM	K	tile	None	water skin temperature
SW	IM	psu	tile	None	water skin salinity
HI	IM	kg	tile	None	seaice skin layer mass
SI	IM	psu	tile	None	seaice skin salinity
FRACICE	IM	1	tile	None	fractional cover of seaice
TI	IM	K	tile	None	seaice skin temperature
VOLICE	IM	m	tile	None	ice category volume per unit area of grid cell
VOLSNO	IM	m	tile	None	sno category volume per unit area of grid cell
ERGICE	IM	J m^{-2}	tile	None	ice category layer internal energy
ERGSNO	IM	J m^{-2}	tile	None	snow category layer internal energy
TAUAGE	IM	s	tile	None	volume weighted mean ice age
MPOND	IM	m	tile	None	pond volume
DAIDTD	IM	tile	None	ice area tendency dueto dynamics	
DVIDTD	IM	cm day^{-1}	tile	None	ice volume tendency dueto dynamics
TI	IM	K	tile	None	seaice skin temperature
UW	EX	m s^{-1}	tile	None	zonal velocity of surface water
VW	EX	m s^{-1}	tile	None	meridional velocity of surface water
UI	EX	m s^{-1}	tile	None	zonal velocity of surface seaice
VI	EX	m s^{-1}	tile	None	meridional velocity of surface seaice
TILELONS	EX	degrees	tile	None	longitude
TILELATS	EX	degrees	tile	None	latitude
KPAR	EX	m^{-1}	tile	None	PAR extinction coefficient

Short Name	Type	Units	Dims	Vert Loc	Long name
TS_FOUND	EX	K	tile	None	foundation temperature for interface layer
FRACICE	EX	1	tile	None	fractional cover of seaice
TAUXIBOT	EX	N m^{-2}	tile	None	eastward stress at base of ice
TAUYIBOT	EX	N m^{-2}	tile	None	northward stress at base of ice
MOM_3D_MASK	EX				
T	EX				
S	EX				
U	EX				
V	EX				
Z	EX				
RHO	EX				
SSH	EX				
TX	EX				
TY	EX				
MLD	EX				
PSI	EX				
AICE	EX				
HICE	EX				
CICE_2D_MASK	EX				

27.2 RUN – Run method for the Ogcm component

INTERFACE:

```
subroutine RUN ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code:
```

DESCRIPTION:
Periodically refreshes the ozone mixing ratios.

28 Module GEOS_OradGridCompMod – Implements absorption of solar radiation in the ocean.

INTERFACE:

```
module GEOSOradGridCompMod
```

USES:

```
use ESMF
use MAPL_Mod
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

DESCRIPTION:
`GEOS_Orad` is a light-weight gridded component that updates the the solar radiation penetrating the ocean

---

28.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer, optional                  :: RC  ! return code
```

DESCRIPTION:
This version uses the MAPL_GenericSetServices, which sets the Initialize and Finalize services, as well as allocating our instance of a generic state and putting it in the gridded component (GC). Here we only need to set the run method and add the state variable specifications (also generic) to our instance of the generic state. This is the way our true state variables get into the ESMF_State INTERNAL, which is in the MAPL_MetaComp. The import and internal variables are allocated and initialized by generic. Here generic is used for the ocean grid. *STATES:*

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
KPAR_PREV	IN	m^{-1}	xy	None	KPAR previous
KPAR_NEXT	IN	m^{-1}	xy	None	KPAR next
SWHEAT	EX	W m^{-2}	xyz	Center	solar heating rate
KPAR	EX	m^{-1}	xy	None	PAR extinction coefficient
COSZ	IM	1	xy	None	cosine of the solar zenith angle
PENUVR	IM	W m^{-2}	xy	None	net downward penetrating direct UV flux
PENPAR	IM	W m^{-2}	xy	None	net downward penetrating direct PAR flux
PENUVF	IM	W m^{-2}	xy	None	net downward penetrating diffuse UV flux
PENPAF	IM	W m^{-2}	xy	None	net downward penetrating diffuse PAR flux
FROCEAN	IM	1	xy	None	ocean fraction of grid cell
H	IM	dyn-m	xyz	Center	Layer
PENUVR	EX	W m^{-2}	xy	None	net downward penetrating direct UV flux
PENPAR	EX	W m^{-2}	xy	None	net downward penetrating direct PAR flux
PENUVF	EX	W m^{-2}	xy	None	net downward penetrating diffuse UV flux
PENPAF	EX	W m^{-2}	xy	None	net downward penetrating diffuse PAR flux
FROCEAN	EX	1	xy	None	ocean fraction of grid cell
H	EX	dyn-m	xyz	Center	Layer

28.2 RUN – First Run stage for the Orad component

INTERFACE:

```
subroutine RUN ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
  type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
  type(ESMF_State),    intent(inout) :: IMPORT ! Import state
  type(ESMF_State),    intent(inout) :: EXPORT ! Export state
  type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
  integer, optional,   intent(  out) :: RC     ! Error code:
```

DESCRIPTION:
Periodically refreshes the penetrating radiation. Eventually this will do a full radiative transfer calculation.

29 Module GEOS_PChemGridCompMod

DESCRIPTION:
`GEOS_PChem` is a proxy component for Aerochem that implements the specification or simple parameterization of the Aerochem species. It works on three types of species: chemical species (oxygen, nitrous oxide, CFC-11, CFC-12, CFC-22, methane, water vapor), diagnostic species (age-of-air), and aerosols (arbitrary).
Each of the chemical species can be treated in one of two ways: parameterized prediction from tabled zonally-symmetric production and loss (P-L) data, or specifycation from zonally-symmetric values (see Resources section for how to control thios behavior). A single flat file containing both the P-L and climatology data *must* be provided (see Resources section). Aerosols are set to 3-dimensional climatological values. The "age-of-air" is predicted by setting the surface values of this tracer to the to zero and advancing other levels by dt. All of these quantities except water vapor are INTERNAL state variables of `GEOS_PChem`. Water vapor is assumed to be a Friendly Import and `GEOS_PChem` leaves it unmodified below the tropopause, or the 200 hPa level if the tropopause is below this level.
For chemical species, the production rate is tabled directly. For the loss, a rate coefficient is tabled. Using Odd-oxygen O_x as an example, the species are updated as follows:

$$\frac{\partial O_x}{\partial t} = \dot{Q}_o - \kappa_o O_x$$

where O_x is the specific mass of odd oxygen, $\dot{Q}_o$ is the odd oxygen production rate, κ_o is the tabled rate coefficient for odd oxygen loss. This is finite differenced in time as:

$$O_x^{n+1} = \frac{O_x^n + \Delta t \dot{Q}_o}{1 + \Delta t \kappa_o}$$

The component reads the monthly tables of the zonally averaged climatology of concentration and production rates and loss frequencies in intialize. These are saved in the private internal state, which is static. The climatologies are interpolated to the natural locations and updated in the run method, and are kept in INTERNAL, an ESMF state attached to the GEOS GENERIC object in the component. If no restart is specified for the INTERNAL, the species are initialized to zero.
We have added a generalization that allows for a multiple-year climatology. Add pchem_clim_years: nnn pchem_clim: dsn to the AGCM.tmpl, where nnn is the number of years in the climatology and dsn is the data set name. We do not allow use of the production and loss parameterization when pchem_clim_years is greater than one.
Ozone is diagnosed from O_x by assuming that it accounts for all O_x at pressures greater than 100 Pa (1 hPa) during the day and at all pressures at night. For those daylit cells where pressures are less than 1 hPa, we assume that the ozone fraction in O_x decreases exponentially with decreasing pressure.

Aerosols are read from 3-dimensional data files that have to be on model levels but may be on any regular lat-lon grid. These are hdf files and horizontal interpolation is done through CFIO. The aerosols are read into a bundle that is exported, and the number and names of the aerosols in the bundle are set from the CFIO file.

PC: ProTeX can parse the list of resources directly from the source code, if the MAPL_GetResouce calls are enclosed within the !BOR/!EOR. See, for example, '*RESOURCES:*' below.

!RESOURCES: ¿¿ RUN_DT: none real seconds Heartbeat. `GEOS_PChem` is called all the time. ¿¿ pchem_clim: 'pchem_clim.dat' string none Zonally-symmetric chemistry data flat file. ¿¿ AEROCLIM: no_aerosols string none Aerosol monthly climatology hdf file. ¿¿ AEROCLIMDEL: no_aerosols string kg kg-1 Aerosol month to month differences hdf file. ¿¿ AEROCLIMYEAR: 2002 integer year Year used in timestamp of the two hdf aerosol files. ¿¿ *name*_FIXED_VALUE: use_file real pppv Constant value at which to fix chemical species *name*. If not specified, the file data will be used. If specified, *name*_RELAXTIME: is ignored. *name* can be any of OX, CH4, N2O, CFC11, CFC12, HCFC22. ¿¿ *name*_RELAXTIME: 0.0 real seconds Timescale of relaxation to climatology on file for chemical species *name*. For values <= 0, the P-L parameterization will be used. To hold at the file's zonally-symmetric climatology, use a small positive number. *name* can be any of OX, CH4, N2O, CFC11, CFC12, HCFC22, H2O. ¿¿ *name*_PCRIT: 1.e+16 real Pa Pressure of level above which the relaxation to climatology is done. This is ignored if *name*_RELAXTIME: is ignored or if *name*_RELAXTIME: is <= 0. *name* can be any of OX, CH4, N2O, CFC11, CFC12, HCFC22. ¿¿ *name*_DELP: 1.e-16 real Pa Pressure interval over which the relaxation to climatology is ramped-in. This is ignored if *name*_RELAXTIME: is ignored or if *name*_RELAXTIME: is <= 0 *name* can be any of OX, CH4, N2O, CFC11, CFC12, HCFC22. ¿¿ *name*_FRIENDLIES: self string none String of colon separated component names to which this species is Friendly. *name* can be any of OX, CH4, N2O, CFC11, CFC12, HCFC22 ¿¿ AOA_FRIENDLIES: 'DYNAMICS:TURBULENCE' string none String of colon separated component names to which Age-of-Air is Friendly. *USES:*

```
use ESMF
use MAPL_Mod
use Chem_Mod
use ESMF_CFIOFileMOD
use MAPL_CFIOMOD
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

29.1 SetServices

DESCRIPTION:
Sets Initialize and Run services.

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer, optional                  :: RC  ! return code
```

STATES:

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
PLE	IM	Pa	xyz	Edge	air pressure
Q	IM	kg kg^{-1}	xyz	Center	specific humidity
TROPP	IM	Pa	xy	None	tropopause pressure
OX	IN	mol mol^{-1}	xyz	Center	odd oxygen volume mixing ratio
N2O	IN	mol mol^{-1}	xyz	Center	nitrous oxide volume mixing ratio
CFC11	IN	mol mol^{-1}	xyz	Center	CFC11 (CCl3F) volume mixing ratio
CFC12	IN	mol mol^{-1}	xyz	Center	CFC12 (CCl2F2) volume mixing ratio
HCFC22	IN	mol mol^{-1}	xyz	Center	HCFC22 (CHClF2) volume mixing ratio
CH4	IN	mol mol^{-1}	xyz	Center	methane volume mixing ratio
AOA	IN	days	xyz	Center	age of air
AERO	IN	kg kg^{-1}	xyz	Center	aerosol mass mixing ratios
AEROTEND	IN	kg kg^{-1} s^{-1}	xyz	Center	aerosol mass mixing ratio tendencies
OX_TEND	EX	mol mol^{-1} s^{-1}	xyz	Center	tendency of odd oxygen mixing ratio due to chemistry
H2O_TEND	EX	kg kg^{-1} s^{-1}	xyz	Center	tendency of water vapor mixing ratio due to chemistry
OX_PROD	EX	mol mol^{-1} s^{-1}	xyz	Center	tendency of odd oxygen volume mixing ratio due to production
OX_LOSS	EX	mol mol^{-1} s^{-1}	xyz	Center	tendency of odd oxygen volume mixing ratio due to loss

Short Name	Type	Units	Dims	Vert Loc	Long name
N2O_PROD	EX	mol mol^{-1} s^{-1}	xyz	Center	tendency of nitrous oxide volume mixing ratio due to production
N2O_LOSS	EX	mol mol^{-1} s^{-1}	xyz	Center	tendency of nitrous oxide volume mixing ratio due to loss
CFC11_PROD	EX	mol mol^{-1} s^{-1}	xyz	Center	tendency of CFC11 volume mixing ratio due to production
CFC11_LOSS	EX	mol mol^{-1} s^{-1}	xyz	Center	tendency of CFC11 volume mixing ratio due to loss
CFC12_PROD	EX	mol mol^{-1} s^{-1}	xyz	Center	tendency of CFC12 volume mixing ratio due to production
CFC12_LOSS	EX	mol mol^{-1} s^{-1}	xyz	Center	tendency of CFC12 volume mixing ratio due to loss
HCFC22_PROD	EX	mol mol^{-1} s^{-1}	xyz	Center	tendency of HCFC22 volume mixing ratio due to production
HCFC22_LOSS	EX	mol mol^{-1} s^{-1}	xyz	Center	tendency of HCFC22 volume mixing ratio due to loss
CH4_PROD	EX	mol mol^{-1} s^{-1}	xyz	Center	tendency of methane volume mixing ratio due to production
CH4_LOSS	EX	mol mol^{-1} s^{-1}	xyz	Center	tendency of methane volume mixing ratio due to loss
H2O_PROD	EX	s^{-1}	xyz	Center	tendency of specific humidity due to production
H2O_LOSS	EX	s^{-1}	xyz	Center	tendency of specific humidity due to loss
O3	EX	kg kg^{-1}	xyz	Center	ozone mass mixing ratio
O3PPMV	EX	ppmv	xyz	Center	ozone volume mixing ratio
TO3	EX	Dobsons	xy	None	total column ozone
TTO3	EX	Dobsons	xy	None	tropospheric column ozone
DUST	EX	kg kg^{-1}	xyz	Center	mineral dust mixing ratio
SALT	EX	kg kg^{-1}	xyz	Center	sea salt mixing ratio
SO4	EX	kg kg^{-1}	xyz	Center	sulfate aerosol mixing ratio
BC	EX	kg kg^{-1}	xyz	Center	black carbon aerosol mixing ratio
OC	EX	kg kg^{-1}	xyz	Center	organic carbon aerosol mixing ratio
AERO_DP	EX	kg m^{-2} s^{-1}	xy		aerosol deposition

29.2 INITIALIZE

DESCRIPTION:

The Initialize method of Pchem gridded component. It reads the production-loss file, which by default is `pchem_clim.dat`, but can be overridden from the configuration. This version reads zonal-mean monthly climatologies and interpolates them to the latitudes of the component's natural grid, which is the inherited grid.

INTERFACE:

```
subroutine Initialize ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code
```

RESOURCES:

Name	Description	Units	Default
'AEROCLIM:'			""
'AEROCLIMDEL:'			""

29.3 RUN

DESCRIPTION:
Updates mixing ratios of Ox, CH4, N2O, CFC11, CFC12, HCFC22. For each species, it either updates a state variable based on tabled production rates and loss times or based on tabled climatological values. The latter is performed when a non-zero relaxation time for the species is found in the configuration. The relaxation times (in seconds) can be specified with the labels XX_RELAXTIME:, for example CFC12_RELAXTIME:. The default is zero, which results in doing a production-loss calculation. To fix the species to climatology, simply specify a very short relaxation time. Since the udpate is done implicitly, this can be done safely.

INTERFACE:

```
subroutine RUN ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code:
```

30 Module GEOS_PhysicsGridCompMod – A Module to combine Short-Wave, Long-Wave Radiation Moist-Physics and Turbulence Gridded Components

INTERFACE:

```
module GEOSPhysicsGridCompMod
```

USES:

```
use ESMF
use MAPL_Mod
use m_chars,  only: uppercase

use GEOS_SurfaceGridCompMod,    only : SurfSetServices       = > SetServices
use GEOS_MoistGridCompMod,      only : MoistSetServices      = > SetServices
use GEOS_TurbulenceGridCompMod, only : TurblSetServices      = > SetServices
use GEOS_RadiationGridCompMod,  only : RadiationSetServices  = > SetServices
use GEOS_ChemGridCompMod,       only : AChemSetServices      = > SetServices
use GEOS_GwdGridCompMod,        only : GwdSetServices        = > SetServices

PGI Module that contains the initialization
routines for the GPUs
use cudafor
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices

= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
```

DESCRIPTION:
This gridded component (GC) combines the Radiation (Short-Wave and Long-Wave), Moist-Physics, Chem, Surface and Turbulence GCs into a new composite Physics GC. The Export Couplings of the Physics GC are the union of the Export Couplings of the individual child

GCs, plus the combined tendencies needed by the dynamics. These last are the pressure-weighted tendencies of the atmospheric state variables U,V,T (due to external diabatic forcing), the tendency of the edge pressures, and a collection of "Friendly" tracers for advection. In the current version, the only friendly tracers are variables from Moist-Physics and Chem.

```
DUDT .... Mass-Weighted U-Wind      Tendency (Pa m /s)
DVDT .... Mass-Weighted V-Wind      Tendency (Pa m /s)
DPEDT ... Edge-Pressure             Tendency (Pa   /s)
DTDT .... Mass-Weighted Temperature Tendency (Pa K /s)
TRACER .. Friendly Tracers                   (unknown)
```

30.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer,             intent(  OUT) :: RC  ! return code
```

DESCRIPTION:
The SetServices for the Physics GC needs to register its Initialize and Run. It uses the MAPL_Generic construct for defining state specs and couplings among its children. In addition, it creates the children GCs (SURF, CHEM, RADIATION, MOIST, TURBULENCE) and runs their respective SetServices. *STATES:*

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
U	IM	$m\ s^{-1}$	xyz	Center	eastward wind
V	IM	$m\ s^{-1}$	xyz	Center	northward wind
DZ	IM	m	xy	None	surface layer height
TH	IM	K	xyz	Center	potential temperature
T	IM	K	xyz	Center	air temperature
S	IM	$m^2\ s^{-2}$	xyz	Center	dry static energy
ZLE	IM	m	xyz	Edge	geopotential height
PLE	IM	Pa	xyz	Edge	air pressure

Short Name	Type	Units	Dims	Vert Loc	Long name
DTDT	EX	Pa K s^{-1}	xyz	Center	pressure weighted tendency of air temperature due to physics
DTDTTOT	EX	K s^{-1}	xyz	Center	tendency of air temperature due to physics
DTDTRAD	EX	K s^{-1}	xyz	Center	tendency of air temperature due to radiation
DUDT	EX	m s^{-2}	xyz	Center	tendency of eastward wind due to physics
DVDT	EX	m s^{-2}	xyz	Center	tendency of northward wind due to physics
DPEDT	EX	Pa s^{-1}	xyz	Edge	tendency of pressure at layer edges due to physics
DMDT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated mass tendency due to physics
THIM	EX	Pa K s^{-1}	xyz	Center	pressure weighted tendency of potential temperature due to moist processes
TIM	EX	K s^{-1}	xyz	Center	tendency of air temperature due to moist processes
TIMFRIC	EX	K s^{-1}	xyz	Center	tendency of air temperature due to moist processes friction
SIT	EX	Pa m^2 s^{-3}	xyz	Center	pressure weighted tendency of dry static energy due to turbulence
TIT	EX	K s^{-1}	xyz	Center	tendency of air temperature due to turbulence
UIT	EX	m s^{-2}	xyz	Center	tendency of eastward wind due to turbulence
VIT	EX	m s^{-2}	xyz	Center	tendency of northward wind due to turbulence
QVIT	EX	kg kg^{-1} s^{-1}	xyz	Center	tendency of specific humidity due to turbulence
QLLSIT	EX	kg kg^{-1} s^{-1}	xyz	Center	tendency of liquid condensate due to turbulence
QILSIT	EX	kg kg^{-1} s^{-1}	xyz	Center	tendency of frozen condensate due to turbulence
OXIT	EX	kg kg^{-1} s^{-1}	xyz	Center	tendency of odd oxygen due to turbulence
OXIM	EX	kg kg^{-1} s^{-1}	xyz	Center	tendency of odd oxygen due to moist processes
TIF	EX	K s^{-1}	xyz	Center	tendency of air temperature due to friction

Short Name	Type	Units	Dims	Vert Loc	Long name
TRADV	EX	X			advected quantities
FTB	EX	W m^{-2}	xyz	Edge	upward net turbulence heat flux
FTU	EX	m^2 s^{-2}	xyz	Edge	upward net turbulence eastward momentum flux
FTV	EX	m^2 s^{-2}	xyz	Edge	upward net turbulence northward momentum flux
TRANA	EX	X			analyzed quantities
KEPHY	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency across physics
PEPHY	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency across physics
PERAD	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency across radiation
PETRB	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency across turbulence
PEMST	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency across moist
PEFRI	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency due to friction
PEGWD	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency across gwd
PECUF	EX	W m^{-2}	xy	None	vertically integrated potential energy tendency due to cumulus friction
DQVDTTRBINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated water vapor tendency due to turbulence
DQVDTMSTINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated water vapor tendency due to moist processes
DQVDTCHMINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated water vapor tendency due to chemistry
DQLDTMSTINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated liquid water tendency due to moist processes

Short Name	Type	Units	Dims	Vert Loc	Long name
DQIDTMSTINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated ice tendency due to moist processes
DOXDTCHMINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated odd oxygen tendency due to chemistry
DQVDTPHYINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated water vapor tendency due to physics
DQLDTPHYINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated liquid water tendency due to physics
DQIDTPHYINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated ice tendency due to physics
DOXDTPHYINT	EX	kg m^{-2} s^{-1}	xy	None	vertically integrated odd oxygen tendency due to physics
DQVDTSCL	EX	kg m^{-2} s^{-1}	xyz	Center	tendency of water vapor due to mass scaling
DQLDTSCL	EX	kg m^{-2} s^{-1}	xyz	Center	tendency of cloud water due to mass scaling
DQIDTSCL	EX	kg m^{-2} s^{-1}	xyz	Center	tendency of cloud ice due to mass scaling
O3PPMV	EX				
OX	EX				
Q	EX				
QCTOT	EX				
U10M	EX				
V10M	EX				
U10N	EX				
V10N	EX				
SNOMAS	EX				
WET1	EX				
TSOIL1	EX				
LWI	EX				
TS	EX				
FRLAND	EX				
FRLANDICE	EX				
FRLAKE	EX				
FROCEAN	EX				
FRACI	EX				
Z0	EX				

30.2 Run – Run method for the composite Physics Gridded Component

INTERFACE:

```
subroutine Run ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code
```

DESCRIPTION:
The run method for the physics calls the children's run methods. It also prepares inputs and couplings amongst them. Its main outputs are the combined tendencies needed by the dynamics.

31 Module GEOS_RadiationGridCompMod–Container for atmospheric radiation calculations

DESCRIPTION:
A Composite MAPL/ESMF gridded component (GC) containing the longwave and shortwave radiation GCs. It is intended as a container for the ESMF/MAPL Solar and Irrad gridded components used in GEOS-5. In SetServices, it creates one instance of each of its two children (SOLAR and IRRAD) Its Run method combines results from the children to produce total radiative exports.

It follows the standard rules for composite ESMF/MAPL GCs. It passes the ESMF grid that appears in the gridded component to both children and all their Imports and Exports are assumed to be on this grid. The grid must be present in the GC and properly initialized before Initialize is called, since `GEOS_RadiationGridCompMod` has its own explicitly declared Import state variables.

The restrictions on the grid are that it be 3-dimensional with two horizontal and one vertical dimension and with only the horizontal dimensions decomposed. The vertical dimension is also assumed to the the third dimension of the Fortran arrays and is indexed from the top down. No particular vertical coordinate is assumed, rather the 3-dimensional field of air pressure at the layer interfaces is a required Import.

This module contains only SetServices, Initialize, and Run methods. The Finalize method being defaulted to the MAPL_Generic version. The SetServices method is the only public entity. There are no public types or data.

USES:

```
use ESMF
use MAPL_Mod
use AeroOptPropTableMod

use GEOS_SolarGridCompMod, only : solarSetServices  = > SetServices
use GEOS_IrradGridCompMod, only : irradSetServices  = > SetServices
use GEOS_SatsimGridCompMod, only : satsimSetServices  = > SetServices
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

31.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer, optional                  :: RC  ! return code
```

DESCRIPTION:
`GEOS_RadiationGridCompMod` registers Initialize and Run methods with ESMF. The Finalize method being defaulted to the MAPL_Generic version and automatically registered by MAPL_GenericSetServices. SetServices registers the two children (SOLAR and IRRAD) with MAPL.

Fields in its Import and Export states that are needed or produced by the component are explicitly registered in SetServices and are described in tables in this documentation. The Import and Export States may also contain fields implicitly inherited from the from the children. Finally, a number of child exports are "promoted" to be explicit Exports of `GEOS_RadiationGridCompMod`.

`GEOS_RadiationGridCompMod` has no Internal state.

STATES:

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
PLEINST	IM	Pa	xyz	Edge	air pressure
DTDT	EX	Pa K s^{-1}	xyz	Center	pressure weighted air temperature tendency due to radiation
RADLW	EX	K s^{-1}	xyz	Center	air temperature tendency due to longwave
RADSW	EX	K s^{-1}	xyz	Center	air temperature tendency due to shortwave
RADLWC	EX	K s^{-1}	xyz	Center	air temperature tendency due to longwave for clear skies
RADSWC	EX	K s^{-1}	xyz	Center	air temperature tendency due to shortwave for clear skies
RADSWNA	EX	K s^{-1}	xyz	Center	air temperature tendency due to shortwave no aerosol
RADLWCNA	EX	K s^{-1}	xyz	Center	air temperature tendency due to longwave for clear skies no aerosol
RADSWCNA	EX	K s^{-1}	xyz	Center	air temperature tendency due to shortwave for clear skies no aerosol
RADSRF	EX	W m^{-2}	xy	None	net downwelling radiation at surface
ALW	EX	W m^{-2}	xy	None	linearization of surface upwelling longwave flux
BLW	EX	W m^{-2} K^{-1}	xy	None	linearization of surface upwelling longwave flux
DRPAR	EX				
DFPAR	EX				
DRNIR	EX				
DFNIR	EX				
DRUVR	EX				
DFUVR	EX				
DRPARN	EX				
DFPARN	EX				
DRNIRN	EX				
DFNIRN	EX				
DRUVRN	EX				
DFUVRN	EX				
FCLD	EX				
TAUCLI	EX				
TAUCLW	EX				
LWS	EX				
LWS0	EX				

Short Name	Type	Units	Dims	Vert Loc	Long name
CLDTT	EX				
ALBEDO	EX				

31.2 RUN – Run method for the composite radiation component

INTERFACE:

```
subroutine RUN ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp),  intent(inout) :: GC     ! Gridded component
type(ESMF_State),     intent(inout) :: IMPORT ! Import state
type(ESMF_State),     intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),     intent(inout) :: CLOCK  ! The clock
integer, optional,    intent(  out) :: RC     ! Error code:
```

DESCRIPTION:
Calls the run methods of solar and irrad and combines their fluxes into a single pressure-weighted temperature tendency.

The Children's Run method is called with the Clock that appears in `GEOS_RadiationGridCompMod`'s Run method. It assumes that, on every call to their Run method, both children update their Exports to values consisten with the current time on that Clock.

The code assumes a single Aerosol bundle will go to both children. The bundle is passed to the children with a MAPL call.

32 Module GEOS_SaltwaterGridCompMod – Implements slab saltwater tiles.

DESCRIPTION:
`GEOS_Saltwater` is a light-weight gridded component that updates the skin sub-tiles at saltwater points, be they ocean, estuary, or salt lake. Currently each tile can have only two subtiles, open-water and ice. But the code is easily extensible to multiple ice types.
The component is written with a two stage run method for use with semi-implicit turbulence components. The first run stage computes exchange coefficients for heat, moisture and momentum at each sub-tile and combines these to tile space, accounting for sub tile variability by passing back an effective surface value of the exchanged quantity.
USES:

```
use sfclayer  ! using module that contains sfc layer code
use ESMF
use MAPL_Mod
use GEOS_UtilsMod
use DragCoefficientsMod
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

32.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer, optional                  :: RC  ! return code
```

DESCRIPTION:
This version uses the MAPL_GenericSetServices, which sets the Initialize and Finalize services, as well as allocating our instance of a generic state and putting it in the gridded component (GC). Here we only need to set the run method and add the state variable specifications (also generic) to our instance of the generic state. This is the way our true state variables get into the ESMF_State INTERNAL, which is in the MAPL_MetaComp. The import and internal variables are allocated and initialized by generic. Here generic is used for tiles. *STATES:*

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
EMIS	EX	1	tile	None	surface emissivity
ALBVR	EX	1	tile	None	surface albedo for visible beam
ALBVF	EX	1	tile	None	surface albedo for visible diffuse
ALBNR	EX	1	tile	None	surface albedo for near infrared beam

Short Name	Type	Units	Dims	Vert Loc	Long name
ALBNF	EX	1	tile	None	surface albedo for near infrared diffuse
EVAPOUT	EX	kg m^{-2} s^{-1}	tile	None	evaporation
SUBLIM	EX	kg m^{-2} s^{-1}	tile	None	sublimation
SNOWOCN	EX	kg m^{-2} s^{-1}	tile	None	ocean snowfall
RAINOCN	EX	kg m^{-2} s^{-1}	tile	None	ocean rainfall
SHOUT	EX	W m^{-2}	tile	None	upward sensible heat flux
SHWTR	EX	W m^{-2}	tile	None	open water upward sensible heat flux
SHICE	EX	W m^{-2}	tile	None	sea ice upward sensible heat flux
HLWUP	EX	W m^{-2}	tile	None	surface outgoing longwave flux
LWNDWTR	EX	W m^{-2}	tile	None	open water net downward longwave flux
LWNDICE	EX	W m^{-2}	tile	None	sea ice net downward longwave flux
LWNDSRF	EX	W m^{-2}	tile	None	surface net downward longwave flux
SWNDSRF	EX	W m^{-2}	tile	None	surface net downward shortwave flux
SWNDWTR	EX	W m^{-2}	tile	None	open water net downward shortwave flux
SWNDICE	EX	W m^{-2}	tile	None	sea ice net downward shortwave flux
HLATN	EX	W m^{-2}	tile	None	total latent energy flux
HLATWTR	EX	W m^{-2}	tile	None	open water latent energy flux
HLATICE	EX	W m^{-2}	tile	None	sea ice latent energy flux
TST	EX	K	tile	None	surface skin temperature
QST	EX	kg kg^{-1}	tile	None	surface specific humidity
TH	EX	K	tile	None	turbulence surface temperature
QH	EX	kg kg^{-1}	tile	None	turbulence surface specific humidity
UH	EX	m s^{-1}	tile	None	turbulence surface zonal velocity
VH	EX	m s^{-1}	tile	None	turbulence surface meridional velocity
DELTS	EX	K	tile	None	change of surface skin temperature
DELQS	EX	kg kg^{-1}	tile	None	change of surface specific humidity
CHT	EX	kg m^{-2} s^{-1}	tile	None	surface heat exchange coefficient

Short Name	Type	Units	Dims	Vert Loc	Long name
CMT	EX	kg m^{-2} s^{-1}	tile	None	surface momentum exchange coefficient
CQT	EX	kg m^{-2} s^{-1}	tile	None	surface moisture exchange coefficient
CNT	EX	1	tile	None	neutral drag coefficient
RIT	EX	1	tile	None	surface bulk richardson number
RET	EX	1	tile	None	surface reynolds number
FRACI	EX	1	tile	None	ice covered fraction of tile
PS	EX	Pa	tile	None	surface pressure
GUST	EX	m s^{-1}	tile	None	gustiness
VENT	EX	m s^{-1}	tile	None	surface ventilation velocity
Z0	EX	m	tile	None	surface roughness
Z0H	EX	m	tile	None	surface roughness for heat
MOT2M	EX	K	tile	None	temperature
MOQ2M	EX	kg kg^{-1}	tile	None	humidity
MOU2M	EX	m s^{-1}	tile	None	zonal
MOV2M	EX	m s^{-1}	tile	None	meridional
MOT10M	EX	K	tile	None	temperature
MOQ10M	EX	kg kg^{-1}	tile	None	humidity
MOU10M	EX	m s^{-1}	tile	None	zonal
MOV10M	EX	m s^{-1}	tile	None	meridional
MOU50M	EX	m s^{-1}	tile	None	zonal
MOV50M	EX	m s^{-1}	tile	None	meridional
TAUXW	EX	N m^{-2}	tile	None	eastward stress over water
TAUYW	EX	N m^{-2}	tile	None	northward stress over water
TAUXI	EX	N m^{-2}	tile	None	eastward stress over ice
TAUYI	EX	N m^{-2}	tile	None	northward stress over ice
TAUXO	EX	N m^{-2}	tile	None	eastward stress on ocean
TAUYO	EX	N m^{-2}	tile	None	northward stress on ocean
OUSTAR3	EX	m^3 s^{-3}	tile	None	ocean ustar cubed
PENUVF	EX	W m^{-2}	tile	None	downwelling uvr diffuse flux at skin base
PENUVR	EX	W m^{-2}	tile	None	downwelling uvr direct flux at skin base
PENPAF	EX	W m^{-2}	tile	None	downwelling par diffuse flux at skin base
PENPAR	EX	W m^{-2}	tile	None	downwelling par direct flux at skin base
DCOOL	EX	m	tile	None	depth of cool layer
DWARM	EX	m	tile	None	depth at base of warm layer
TDROP	EX	K	tile	None	temperature drop across cool layer
QCOOL	EX	W m^{-2}	tile	None	net heating in cool layer

Short Name	Type	Units	Dims	Vert Loc	Long name
QWARM	EX	W m^{-2}	tile	None	net heating in warm layer
SWCOOL	EX	W m^{-2}	tile	None	solar heating in cool layer
SWWARM	EX	W m^{-2}	tile	None	solar heating in warm layer
PHIW	EX	1	tile	None	Similarity function in warm layer
LANGM	EX	1	tile	None	Langmuir number
USTARW	EX	m s^{-1}	tile	None	ustar over water
TBAR	EX	K	tile	None	mean temperature of interface layer
LCOOL	EX	1	tile	None	Saunders parameter
BCOOL	EX	m^2 s^{-3}	tile	None	bouyancy generation in cool layer
TDEL	EX	K	tile	None	temperature at base of cool layer
TS_FOUND	EX	K	tile	None	foundation temperature for interface layer
HSKINW	IN	kg m^{-2}	tile	None	water skin layer mass
TSKINW	IN	K	tile	None	water skin temperature
SSKINW	IN	psu	tile	None	water skin salinity
HSKINI	IN	kg m^{-2}	tile	None	ice skin layer mass
TSKINI	IN	K	tile	None	ice skin temperature
SSKINI	IN	psu	tile	None	ice skin salinity
QS	IN	kg kg^{-1}	TileTile	None	surface specific humidity
CH	IN	kg m^{-2} s^{-1}	TileTile	None	surface heat exchange coefficient
CM	IN	kg m^{-2} s^{-1}	TileTile	None	surface momentum exchange coefficient
CQ	IN	kg m^{-2} s^{-1}	TileTile	None	surface moisture exchange coefficient
Z0	IN	m	TileTile	None	aerodynamic roughness
WW	IN	m^2 s^{-2}	TileTile	None	vertical velocity scale squared
TWMTS	IN	K	tile	None	departure of skin temperature from mean interface temperature
DTWARM	IN	K	tile	None	departure of TDEL from TS FOUND
ALW	IM	W m^{-2}	tile	None	linearization of surface upwelling longwave flux
BLW	IM	W m^{-2} K^{-1}	tile	None	linearization of surface upwelling longwave flux
LWDNSRF	IM	W m^{-2}	tile	None	surface downwelling longwave flux

Short Name	Type	Units	Dims	Vert Loc	Long name
DRPAR	IM	W m^{-2}	tile	None	surface downwelling par beam flux
DFPAR	IM	W m^{-2}	tile	None	surface downwelling par diffuse flux
DRNIR	IM	W m^{-2}	tile	None	surface downwelling nir beam flux
DFNIR	IM	W m^{-2}	tile	None	surface downwelling nir diffuse flux
DRUVR	IM	W m^{-2}	tile	None	surface downwelling uvr beam flux
DFUVR	IM	W m^{-2}	tile	None	surface downwelling uvr diffuse flux
EVAP	IM	kg m^{-2} s^{-1}	tile	None	evaporation
SH	IM	W m^{-2}	tile	None	upward sensible heat flux
TAUX	IM	N m^{-2}	tile	None	eastward surface stress
TAUY	IM	N m^{-2}	tile	None	northward surface stress
DEVAP	IM	kg m^{-2} s^{-1}	tile	None	derivative of evaporation
DSH	IM	W m^{-2}	tile	None	derivative of upward sensible heat flux
SNO	IM	kg m^{-2} s^{-1}	tile	None	snowfall
TA	IM	K	tile	None	surface air temperature
QA	IM	kg kg^{-1}	tile	None	surface air specific humidity
UU	IM	m s^{-1}	tile	None	surface wind speed
UWINDLMTILE	IM	m s^{-1}	tile	None	levellm uwind
VWINDLMTILE	IM	m s^{-1}	tile	None	levellm vwind
DZ	IM	m	tile	None	surface layer height
PS	IM	Pa	tile	None	surface pressure
PCU	IM	kg m^{-2} s^{-1}	tile	None	liquid water convective precipitation
PLS	IM	kg m^{-2} s^{-1}	tile	None	liquid water large scale precipitation
THATM	IM	K	tile	None	effective surface skin temperature
QHATM	IM	kg kg^{-1}	tile	None	effective surface specific humidity
UHATM	IM	m s^{-1}	tile	None	effective surface zonal velocity
VHATM	IM	m s^{-1}	tile	None	effective surface meridional velocity
CTATM	IM	kg m^{-2} s^{-1}	tile	None	surface exchange coefficient for heat
CQATM	IM	kg m^{-2} s^{-1}	tile	None	surface exchange coefficient for moisture

Short Name	Type	Units	Dims	Vert Loc	Long name
CMATM	IM	kg m^{-2} s^{-1}	tile	None	surface exchange coefficient for momentum
FRACICE	IM	1	tile	None	ice covered fraction of tile
UW	IM	m s^{-1}	tile	None	zonal velocity of surface water
UI	IM	m s^{-1}	tile	None	zonal velocity of surface ice
VW	IM	m s^{-1}	tile	None	meridional velocity of surface water
VI	IM	m s^{-1}	tile	None	meridional velocity of surface ice
KPAR	IM	m^{-1}	tile	None	PAR extinction coefficient
TS_FOUND	IM	K	tile	None	foundation temperature for interface layer
DTSDT	IM	K s^{-1}	tile	None	skin temperature analysis tendency

---

32.2 RUN1 – First Run stage for the Saltwater component

INTERFACE:

```
subroutine RUN1 ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
  type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
  type(ESMF_State),    intent(inout) :: IMPORT ! Import state
  type(ESMF_State),    intent(inout) :: EXPORT ! Export state
  type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
  integer, optional,   intent(  out) :: RC     ! Error code:
```

DESCRIPTION:
Periodically refreshes the sea-rface conditions

32.3 RUN2 – Second Run stage for the Saltwater component

INTERFACE:

```
subroutine RUN2 ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code:
```

DESCRIPTION:
Periodically refreshes the ozone mixing ratios.

33 Module GEOS_Satsim – A Module to drive satellite simulators using grid mean cloud parameters

DESCRIPTION:
`GEOS_MoistGridCompMod` implements mosit processes in GEOS-5. These include all processes that involve phase changes in the atmosphere, such as large-scale condensation, convective clouds, and all rain and cloud formation. It's state consists of water vapor, various types of condensate, and fractions of various cloud types.
USES:

```
use ESMF
use MAPL_Mod
use GEOS_UtilsMod

use gettau

use MOD_COSP_TYPES
use MOD_GEOS5_COSP_CONST
use MOD_COSP, only: COSP
 use MOD_COSP_Modis_Simulator, only: cosp_modis
use MOD_COSP_Modis_Simulator
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices

Private State
TYPE SatSim_State
   PRIVATE

   integer    :: nmask_vars ! number of masked variables
   character(len = ESMF_MAXSTR), pointer :: export_name(:)  = > null()
   character(len = ESMF_MAXSTR), pointer :: mask_name(:)    = > null()
   character(len = ESMF_MAXSTR), pointer :: newvar_name(:)  = > null()
```

```
      logical, pointer :: newvar(:)  = > null()

   END TYPE SatSim_State

   Hook for ESMF
   -------------

   TYPE SatSim_Wrap
      TYPE(SatSim_State), pointer :: PTR  = > null()
   END TYPE SatSim_Wrap
```

33.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer, optional                  :: RC  ! return code
```

DESCRIPTION:
`GEOS_MoistGridCompMod` uses the default Initialize and Finalize services, but registers its own Run method. *STATES:*

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
T	IM	K	xyz	Center	air temperature
QV	IM	1	xyz	Center	specific humidity
FCLD	IM	1	xyz	Center	cloud area fraction
RL	IM	m	xyz	Center	liquid cloud particle effective radius
RI	IM	m	xyz	Center	ice phase cloud particle effective radius
RS	IM	m	xyz	Center	snow particle effective radius
RR	IM	m	xyz	Center	rain cloud particle effective radius
PLE	IM	Pa	xyz	Edge	Edge pressures
ZLE	IM	m	xyz	Edge	Edge heights

Short Name	Type	Units	Dims	Vert Loc	Long name
QLLS	IM	1	xyz	Center	mass fraction of large scale cloud liquid water
QLCN	IM	1	xyz	Center	mass fraction of convective cloud liquid water
QILS	IM	1	xyz	Center	mass fraction of large scale cloud ice water
QICN	IM	1	xyz	Center	mass fraction of convective cloud ice water
QRTOT	IM	kg kg^{-1}	xyz	Center	mass fraction of falling rain
QSTOT	IM	kg kg^{-1}	xyz	Center	mass fraction of falling snow
TS	IM	K	xy	None	skin temperature
MCOSZ	IM	1	xy	None	mean cosine of the solar zenith angle
FRLAND	IM	1	xy	None	fraction of land
FROCEAN	IM	1	xy	None	fraction of ocean
ISCCP_CU_UA	EX	1	xy	None	fraction of thin lower cumulus
ISCCP_CU_UB	EX	1	xy	None	fraction of thick lower cumulus
ISCCP_CU_OA	EX	1	xy	None	fraction of thin higher cumulus
ISCCP_CU_OB	EX	1	xy	None	fraction of thick higher cumulus
ISCCP_STCU_UA	EX	1	xy	None	fraction of thin lower stratocumulus
ISCCP_STCU_UB	EX	1	xy	None	fraction of thick lower stratocumulus
ISCCP_STCU_OA	EX	1	xy	None	fraction of thin higher stratocumulus
ISCCP_STCU_OB	EX	1	xy	None	fraction of thick higher stratocumulus
ISCCP_ST_UA	EX	1	xy	None	fraction of thin lower stratus
ISCCP_ST_UB	EX	1	xy	None	fraction of thick lower stratus
ISCCP_ST_OA	EX	1	xy	None	fraction of thin higher stratus
ISCCP_ST_OB	EX	1	xy	None	fraction of thick higher stratus
ISCCP_ACU_UA	EX	1	xy	None	fraction of thin lower altocumulus
ISCCP_ACU_UB	EX	1	xy	None	fraction of thick lower altocumulus
ISCCP_ACU_OA	EX	1	xy	None	fraction of thin higher altocumulus

Short Name	Type	Units	Dims	Vert Loc	Long name
ISCCP_ACU_OB	EX	1	xy	None	fraction of thick higher altocumulus
ISCCP_AST_UA	EX	1	xy	None	fraction of thin lower altostratus
ISCCP_AST_UB	EX	1	xy	None	fraction of thick lower altostratus
ISCCP_AST_OA	EX	1	xy	None	fraction of thin higher altostratus
ISCCP_AST_OB	EX	1	xy	None	fraction of thick higher altostratus
ISCCP_NST_UA	EX	1	xy	None	fraction of thin lower nimbostratus
ISCCP_NST_UB	EX	1	xy	None	fraction of thick lower nimbostratus
ISCCP_NST_OA	EX	1	xy	None	fraction of thin higher nimbostratus
ISCCP_NST_OB	EX	1	xy	None	fraction of thick higher nimbostratus
ISCCP_CI_UA	EX	1	xy	None	fraction of thin lower cirrus
ISCCP_CI_UB	EX	1	xy	None	fraction of thick lower cirrus
ISCCP_CI_MA	EX	1	xy	None	fraction of thin middle cirrus
ISCCP_CI_MB	EX	1	xy	None	fraction of thick middle cirrus
ISCCP_CI_OA	EX	1	xy	None	fraction of thin higher cirrus
ISCCP_CI_OB	EX	1	xy	None	fraction of thick higher cirrus
ISCCP_CIST_UA	EX	1	xy	None	fraction of thin lower cirrostratus
ISCCP_CIST_UB	EX	1	xy	None	fraction of thick lower cirrostratus
ISCCP_CIST_MA	EX	1	xy	None	fraction of thin middle cirrostratus
ISCCP_CIST_MB	EX	1	xy	None	fraction of thick middle cirrostratus
ISCCP_CIST_OA	EX	1	xy	None	fraction of thin higher cirrostratus
ISCCP_CIST_OB	EX	1	xy	None	fraction of thick higher cirrostratus
ISCCP_CB_UA	EX	1	xy	None	fraction of thin lower cumulonimbus
ISCCP_CB_UB	EX	1	xy	None	fraction of thick lower cumulonimbus

Short Name	Type	Units	Dims	Vert Loc	Long name
ISCCP_CB_MA	EX	1	xy	None	fraction of thin middle cumulonimbus
ISCCP_CB_MB	EX	1	xy	None	fraction of thick middle cumulonimbus
ISCCP_CB_OA	EX	1	xy	None	fraction of thin higher cumulonimbus
ISCCP_CB_OB	EX	1	xy	None	fraction of thick higher cumulonimbus
ISCCP_SUBV1	EX	1	xy	None	fraction of subvisible cloud 0 180 hPa
ISCCP_SUBV2	EX	1	xy	None	fraction of subvisible cloud 180 310 hPa
ISCCP_SUBV3	EX	1	xy	None	fraction of subvisible cloud 310 440 hPa
ISCCP_SUBV4	EX	1	xy	None	fraction of subvisible cloud 440 560 hPa
ISCCP_SUBV5	EX	1	xy	None	fraction of subvisible cloud 560 680 hPa
ISCCP_SUBV6	EX	1	xy	None	fraction of subvisible cloud 680 800 hPa
ISCCP_SUBV7	EX	1	xy	None	fraction of subvisible cloud 800 SFC hPa
ISCCP1	EX	1	xy	None	isccp output 0 180 hPa
ISCCP2	EX	1	xy	None	isccp output 180 310 hPa
ISCCP3	EX	1	xy	None	isccp output 310 440 hPa
ISCCP4	EX	1	xy	None	isccp output 440 560 hPa
ISCCP5	EX	1	xy	None	isccp output 560 680 hPa
ISCCP6	EX	1	xy	None	isccp output 680 800 hPa
ISCCP7	EX	1	xy	None	isccp output 800 SFC hPa
TCLISCCP	EX	1	xy	None	isccp total
CTPISCCP	EX	1	xy	None	isccp air pressure at cloud top
ALBISCCP	EX	1	xy	None	isccp cloud albedo
SGFCLD	EX	1	xyz	Center	summed subgrid cloud fraction from scops
LIDARPMOL	EX	m^{-1} sr^{-1}	xyz	Center	molecular attenuated backscatter lidar signal power
LIDARPTOT	EX	m^{-1} sr^{-1}	xyz	Center	total attenuated backscatter lidar signal power
LIDARTAUTOT	EX	1	xyz	Center	optical thickess integrated from top to level z
RADARZETOT	EX	1	xyz	Center	radar
CLCALIPSO	EX	1	xyz	Center	calipso total cloud fraction

Short Name	Type	Units	Dims	Vert Loc	Long name
CLLCALIPSO	EX	1	xy	None	calipso low level cloud fraction
CLMCALIPSO	EX	1	xy	None	calipso mid level cloud fraction
CLHCALIPSO	EX	1	xy	None	calipso high level cloud fraction
CLTCALIPSO	EX	1	xy	None	calipso total cloud fraction
PARASOLREFL0	EX	1	xy	None	parasol reflectance
PARASOLREFL1	EX	1	xy	None	parasol reflectance 1
PARASOLREFL2	EX	1	xy	None	parasol reflectance 2
PARASOLREFL3	EX	1	xy	None	parasol reflectance 3
PARASOLREFL4	EX	1	xy	None	parasol reflectance 4
PARASOLREFL5	EX	1	xy	None	parasol reflectance 5
RADARLTCC	EX	1	xy	None	Radar and lidar total cloud amount
CLCALIPSO2	EX	1	xyz	Center	calipsonocloudsat cloud fraction
CFADLIDARSR532_01	EX	1	xyz	Center	calipso scattering ratio cfad 01
CFADLIDARSR532_02	EX	1	xyz	Center	calipso scattering ratio cfad 02
CFADLIDARSR532_03	EX	1	xyz	Center	calipso scattering ratio cfad 03
CFADLIDARSR532_04	EX	1	xyz	Center	calipso scattering ratio cfad 04
CFADLIDARSR532_05	EX	1	xyz	Center	calipso scattering ratio cfad 05
CFADLIDARSR532_06	EX	1	xyz	Center	calipso scattering ratio cfad 06
CFADLIDARSR532_07	EX	1	xyz	Center	calipso scattering ratio cfad 07
CFADLIDARSR532_08	EX	1	xyz	Center	calipso scattering ratio cfad 08
CFADLIDARSR532_09	EX	1	xyz	Center	calipso scattering ratio cfad 09
CFADLIDARSR532_10	EX	1	xyz	Center	calipso scattering ratio cfad 10
CFADLIDARSR532_11	EX	1	xyz	Center	calipso scattering ratio cfad 11
CFADLIDARSR532_12	EX	1	xyz	Center	calipso scattering ratio cfad 12
CFADLIDARSR532_13	EX	1	xyz	Center	calipso scattering ratio cfad 13

Short Name	Type	Units	Dims	Vert Loc	Long name
CFADLIDARSR532_14	EX	1	xyz	Center	calipso scattering ratio cfad 14
CFADLIDARSR532_15	EX	1	xyz	Center	calipso scattering ratio cfad 15
CLOUDSATCFAD01	EX	1	xyz	Center	cloudsat radar reflectivity cfad
CLOUDSATCFAD02	EX	1	xyz	Center	cloudsat radar reflectivity cfad
CLOUDSATCFAD03	EX	1	xyz	Center	cloudsat radar reflectivity cfad
CLOUDSATCFAD04	EX	1	xyz	Center	cloudsat radar reflectivity cfad
CLOUDSATCFAD05	EX	1	xyz	Center	cloudsat radar reflectivity cfad
CLOUDSATCFAD06	EX	1	xyz	Center	cloudsat radar reflectivity cfad
CLOUDSATCFAD07	EX	1	xyz	Center	cloudsat radar reflectivity cfad
CLOUDSATCFAD08	EX	1	xyz	Center	cloudsat radar reflectivity cfad
CLOUDSATCFAD09	EX	1	xyz	Center	cloudsat radar reflectivity cfad
CLOUDSATCFAD10	EX	1	xyz	Center	cloudsat radar reflectivity cfad
CLOUDSATCFAD11	EX	1	xyz	Center	cloudsat radar reflectivity cfad
CLOUDSATCFAD12	EX	1	xyz	Center	cloudsat radar reflectivity cfad
CLOUDSATCFAD13	EX	1	xyz	Center	cloudsat radar reflectivity cfad
CLOUDSATCFAD14	EX	1	xyz	Center	cloudsat radar reflectivity cfad
CLOUDSATCFAD15	EX	1	xyz	Center	cloudsat radar reflectivity cfad
MDSCLDFRCTTL	EX	1	xy	None	modis cloud fraction total mean
MDSCLDFRCWTR	EX	1	xy	None	modis cloud fraction water mean
MDSCLDFRCH2O	EX	1	xy	None	modis cloud fraction water mean
MDSCLDFRCICE	EX	1	xy	None	modis cloud fraction ice mean
MDSCLDFRCHI	EX	1	xy	None	modis cloud fraction high mean
MDSCLDFRCMID	EX	1	xy	None	modis cloud fraction mid mean

Short Name	Type	Units	Dims	Vert Loc	Long name
MDSCLDFRCLO	EX	1	xy	None	modis cloud fraction low mean
MDSOPTHCKTTL	EX	1	xy	None	modis optical thickness total mean
MDSOPTHCKWTR	EX	1	xy	None	modis optical thickness water mean
MDSOPTHCKH2O	EX	1	xy	None	modis optical thickness water mean
MDSOPTHCKICE	EX	1	xy	None	modis optical thickness ice mean
MDSOPTHCKTTLLG	EX	1	xy	None	modis optical thickness total logmean
MDSOPTHCKWTRLG	EX	1	xy	None	modis optical thickness water logmean
MDSOPTHCKH2OLG	EX	1	xy	None	modis optical thickness water logmean
MDSOPTHCKICELG	EX	1	xy	None	modis optical thickness ice logmean
MDSCLDSZWTR	EX	1	xy	None	modis cloud particle size water mean
MDSCLDSZH2O	EX	1	xy	None	modis cloud particle size water mean
MDSCLDSZICE	EX	1	xy	None	modis cloud particle size ice mean
MDSCLDTOPPS	EX	1	xy	None	modis cloud top pressure total mean
MDSWTRPATH	EX	1	xy	None	modis liquid water path mean
MDSH2OPATH	EX	1	xy	None	modis liquid water path mean
MDSICEPATH	EX	1	xy	None	modis ice water path mean
MDSTAUPRSHIST11	EX	1	xy	None	modis tau pressure histogram bin 1 1
MDSTAUPRSHIST12	EX	1	xy	None	modis tau pressure histogram bin 1 2
MDSTAUPRSHIST13	EX	1	xy	None	modis tau pressure histogram bin 1 3
MDSTAUPRSHIST14	EX	1	xy	None	modis tau pressure histogram bin 1 4
MDSTAUPRSHIST15	EX	1	xy	None	modis tau pressure histogram bin 1 5
MDSTAUPRSHIST16	EX	1	xy	None	modis tau pressure histogram bin 1 6
MDSTAUPRSHIST17	EX	1	xy	None	modis tau pressure histogram bin 1 7

Short Name	Type	Units	Dims	Vert Loc	Long name
MDSTAUPRSHIST21	EX	1	xy	None	modis tau pressure histogram bin 2 1
MDSTAUPRSHIST22	EX	1	xy	None	modis tau pressure histogram bin 2 2
MDSTAUPRSHIST23	EX	1	xy	None	modis tau pressure histogram bin 2 3
MDSTAUPRSHIST24	EX	1	xy	None	modis tau pressure histogram bin 2 4
MDSTAUPRSHIST25	EX	1	xy	None	modis tau pressure histogram bin 2 5
MDSTAUPRSHIST26	EX	1	xy	None	modis tau pressure histogram bin 2 6
MDSTAUPRSHIST27	EX	1	xy	None	modis tau pressure histogram bin 2 7
MDSTAUPRSHIST31	EX	1	xy	None	modis tau pressure histogram bin 3 1
MDSTAUPRSHIST32	EX	1	xy	None	modis tau pressure histogram bin 3 2
MDSTAUPRSHIST33	EX	1	xy	None	modis tau pressure histogram bin 3 3
MDSTAUPRSHIST34	EX	1	xy	None	modis tau pressure histogram bin 3 4
MDSTAUPRSHIST35	EX	1	xy	None	modis tau pressure histogram bin 3 5
MDSTAUPRSHIST36	EX	1	xy	None	modis tau pressure histogram bin 3 6
MDSTAUPRSHIST37	EX	1	xy	None	modis tau pressure histogram bin 3 7
MDSTAUPRSHIST41	EX	1	xy	None	modis tau pressure histogram bin 4 1
MDSTAUPRSHIST42	EX	1	xy	None	modis tau pressure histogram bin 4 2
MDSTAUPRSHIST43	EX	1	xy	None	modis tau pressure histogram bin 4 3
MDSTAUPRSHIST44	EX	1	xy	None	modis tau pressure histogram bin 4 4
MDSTAUPRSHIST45	EX	1	xy	None	modis tau pressure histogram bin 4 5
MDSTAUPRSHIST46	EX	1	xy	None	modis tau pressure histogram bin 4 6
MDSTAUPRSHIST47	EX	1	xy	None	modis tau pressure histogram bin 4 7

Short Name	Type	Units	Dims	Vert Loc	Long name
MDSTAUPRSHIST51	EX	1	xy	None	modis tau pressure histogram bin 5 1
MDSTAUPRSHIST52	EX	1	xy	None	modis tau pressure histogram bin 5 2
MDSTAUPRSHIST53	EX	1	xy	None	modis tau pressure histogram bin 5 3
MDSTAUPRSHIST54	EX	1	xy	None	modis tau pressure histogram bin 5 4
MDSTAUPRSHIST55	EX	1	xy	None	modis tau pressure histogram bin 5 5
MDSTAUPRSHIST56	EX	1	xy	None	modis tau pressure histogram bin 5 6
MDSTAUPRSHIST57	EX	1	xy	None	modis tau pressure histogram bin 5 7
MDSTAUPRSHIST61	EX	1	xy	None	modis tau pressure histogram bin 6 1
MDSTAUPRSHIST62	EX	1	xy	None	modis tau pressure histogram bin 6 2
MDSTAUPRSHIST63	EX	1	xy	None	modis tau pressure histogram bin 6 3
MDSTAUPRSHIST64	EX	1	xy	None	modis tau pressure histogram bin 6 4
MDSTAUPRSHIST65	EX	1	xy	None	modis tau pressure histogram bin 6 5
MDSTAUPRSHIST66	EX	1	xy	None	modis tau pressure histogram bin 6 6
MDSTAUPRSHIST67	EX	1	xy	None	modis tau pressure histogram bin 6 7
MDSTAUPRSHIST71	EX	1	xy	None	modis tau pressure histogram bin 7 1
MDSTAUPRSHIST72	EX	1	xy	None	modis tau pressure histogram bin 7 2
MDSTAUPRSHIST73	EX	1	xy	None	modis tau pressure histogram bin 7 3
MDSTAUPRSHIST74	EX	1	xy	None	modis tau pressure histogram bin 7 4
MDSTAUPRSHIST75	EX	1	xy	None	modis tau pressure histogram bin 7 5
MDSTAUPRSHIST76	EX	1	xy	None	modis tau pressure histogram bin 7 6
MDSTAUPRSHIST77	EX	1	xy	None	modis tau pressure histogram bin 7 7
MISRMNCLDTP	EX	m	xy	None	MISR mead cloud top height

Short Name	Type	Units	Dims	Vert Loc	Long name
MISRCLDAREA	EX	1	xy	None	MISR cloud area
MISRLYRTP0	EX	1	xy	None	MISR layer top
MISRLYRTP250	EX	1	xy	None	MISR layer top
MISRLYRTP750	EX	1	xy	None	MISR layer top
MISRLYRTP1250	EX	1	xy	None	MISR layer top
MISRLYRTP1750	EX	1	xy	None	MISR layer top
MISRLYRTP2250	EX	1	xy	None	MISR layer top
MISRLYRTP2750	EX	1	xy	None	MISR layer top
MISRLYRTP3500	EX	1	xy	None	MISR layer top
MISRLYRTP4500	EX	1	xy	None	MISR layer top
MISRLYRTP6000	EX	1	xy	None	MISR layer top
MISRLYRTP8000	EX	1	xy	None	MISR layer top
MISRLYRTP10000	EX	1	xy	None	MISR layer top
MISRLYRTP12000	EX	1	xy	None	MISR layer top
MISRLYRTP14000	EX	1	xy	None	MISR layer top
MISRLYRTP16000	EX	1	xy	None	MISR layer top
MISRLYRTP18000	EX	1	xy	None	MISR layer top
MISRFQ0	EX	1	xy	None	MISR cloud area
MISRFQ250	EX	1	xy	None	MISR cloud area
MISRFQ750	EX	1	xy	None	MISR cloud area
MISRFQ1250	EX	1	xy	None	MISR cloud area
MISRFQ1750	EX	1	xy	None	MISR cloud area
MISRFQ2250	EX	1	xy	None	MISR cloud area
MISRFQ2750	EX	1	xy	None	MISR cloud area
MISRFQ3500	EX	1	xy	None	MISR layer top
MISRFQ4500	EX	1	xy	None	MISR layer top
MISRFQ6000	EX	1	xy	None	MISR layer top
MISRFQ8000	EX	1	xy	None	MISR layer top
MISRFQ10000	EX	1	xy	None	MISR layer top
MISRFQ12000	EX	1	xy	None	MISR layer top
MISRFQ14000	EX	1	xy	None	MISR layer top
MISRFQ16000	EX	1	xy	None	MISR layer top
MISRFQ18000	EX	1	xy	None	MISR layer top
SATORB	IM	days	xy		Satellite orbits

33.2 RUN – Run method for the SATSIM component

INTERFACE:

```
subroutine RUN ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code:
```

DESCRIPTION:
This version uses the MAPL_GenericSetServices. This function sets the Initialize and Finalize services, as well as allocating

34 Module GEOS_SolarGridCompMod – Computes solar radiation fluxes in a cloudy atmosphere

/* DESCRIPTION:
`GEOS_SolarGridCompMod` is an ESMF/MAPL gridded component that performs a broadband calculation of shortwave radiative fluxes for use as a solar radiation parameterization in atmospheric models on a sphere.

Scientific Basis: The radiative transfer calculation is based on M-D Chou shortwave parameterization. The basic reference for the scheme is: Chou and Suarez 1999: A Solar Radiation Parameterization for Atmospheric Studies, NASA-TM-1999-104606, Vol 15. An updated version of this report can be found in SolarDoc.pdf in this directory.

The parameterization treats direct and diffuse fluxes of solar radiation in eight spectral bands:

```
in the uv region :
   index  1 for the 0.225-0.285 micron band
   index  2 for the 0.175-0.225;0.285-0.300 micron band
   index  3 for the 0.300-0.325 micron band
   index  4 for the 0.325-0.4 micron band
in the par region :
   index  5 for the 0.4-0.690 micron band
in the infrared region :
   index  6 for the 0.690-1.220 micron band
   index  7 for the 1.220-2.270 micron band
   index  8 for the 2.270-3.850 micron band
```

It includes gaseous absorption due to water vapor, ozone, CO_2, and molecular oxygen and the effects of molecular scattering, as well as multiple scattering due to clouds and aerosols. It allows clouds to occur in any layer and horizontal cloud cover fractions must be specified for all layers; clear layers simply have a fraction of zero. Vertically, the layers are assumed to be filled by cloud. To simplify the treatment of cloud effects, the model layers, are grouped into three super layers. Effective cloud properties are then parameterized by assuming that clouds are maximally overlapped within the super layers and randomly overlapped between the super layers. The optical properties of cloud particles depend on the liquid, ice, and rain mixing ratios, as well as on spatially dependent effective radii for the three species. These are all inputs to the component.

The parameterization can include the effects of an arbitrary number of aerosol species. Aerosol optical thickness, single-scattering albedo, and asymmetry factor must be determined as functions of height and spectral band for each species.

Code Implementation:

`GEOS_SolarGridCompMod` is an encapsulation of Chou's plug-compatible SORAD Fortran routine in a MAPL/ESMF gridded component (GC). It follows the standard rules for an ESMF/MAPL GCs. It operates on the ESMF grid that appears in the gridded component. This grid must be present in the GC and properly initialized before Initialize is called. The only restrictions on the grid are that it be 3-dimensional with two horizontal and one vertical dimension and with only the horizontal dimensions decomposed. The vertical dimension is also assumed to the the thrid dimension of the Fortran arrays and is indexed from the top down. No particular vertical coordinate is assumed, rather the 3-dimensional field of air pressure at the layer interfaces is a required Import.

This module contains only SetServices and Run methods. The Initialize and Finalize methods being defaulted to the MAPL_Generic versions. The SetServices method is the only public entity. There are no public types or data.

The contents of the Import, Export, and Internal States are explicitly described in SetServices and in tables in this documentation. All quantities in these states are in either ESMF Fields or Bundles, and all share a common grid—the ESMF grid in the gridded component at the time Initialize (MAPL_GenericInitialize, in this case) was called. All outputs appearing in the Export state are optional and are filled only if they have been allocated. All filled Exports are valid for the time interval on the GC's clock when the run method is invoked. Imports can be from either an instantaneous or a time-averaged state of the atmosphere. All Imports are read-only; none are Friendly. Most imports are simple ESMF Fields containing 2- or 3-dimensional quantities, such as temperature and humidity, needed in the flux calculation. Non-cloud aerosol amounts are the exception; they appear in an ESMF Bundle.

The net (+ve downward) fluxes on the Export state are defined at the layer interfaces, which are indexed from the top of the atmosphere (L = 0) to the surface. Incident fluxes at the surface also appear in the Export state; these are separated into direct (beam) and diffuse fluxes for three spectral bands (uv, par, nir), as defined in the table above.

The full transfer calculation is done infrequently and its results kept in the Internal state. The frequency of full calculations is controlled by an alarm whose interval can be set from a value in the configuration and whose origin is taken as the beginning of the run. For the full calculations, solar fluxes are computed based on mean zenith angles averaged over sun positions for a given period (the long interval, which can be specified in the configuration) beyond the current time on the input clock. On every call to the Run method, whatever the state of the alarm that controls the full calculation, the sun's position is updated to the mean position for the clock's current interval and fluxes are updated based on normalized fluxes computed during the previous full transfer calculation, but using the TOA insolation for the current time on the clock. Because of this intermittent scheme, checkpoint-restart sequences are seamless only when interrupted at the time of the full calculation.

The calculation relies in MAPL's Astronomy layer, which in turn assumes that the ESMF grid can be queried for latitude and longitude coordinates.

Configuration:

Like all MAPL GCs, `GEOS_SolarGridCompMod` assumes that the configuration in the ESMF GC is open and treats it as an environment from which it can *at any time* read control information. It uses MAPL rules for scanning this configuration.

VARIABLE	DESCRIPTION	UNITS	DEFAULT	NOTES
RUN_DT:	Short time interval	(seconds)	none	
DT:	Long time interval	(seconds)	RUN_DT	
AVGR:	Averaging interval	(seconds)	DT	
PRS_LOW_MID_CLOUDS:	Interface pressure between the low and middle cloud layers	(Pa)	70000.	
PRS_MID_HIGH_CLOUDS:	Interface pressure between the high and middle cloud layers	(Pa)	40000.	
SOLAR_CONSTANT:		(W m-2)	none	Use -1 for time-dependent val
CO2:	CO2 concentration	(ppmv)	none	Use -1 for time-dependent val

BUGS:

- Aerosol properties for each aerosol in the Bundle are obtained by calling a global method (Get_AeroOptProp) that must recognize the aerosol by its Field name in the Bundle. This is a placeholder for a scheme in which each Field carries with it a method for computing its aerosol's optical properties.
- The grid must have two horizontal dimensions and they must be the inner dimensions of Fortran arrays.

- The load-balancing relies on the grid describing a sphere. Everything works for non-spherical grids but the load-balancing should be disabled and this can be done only by going into the code.

  */

USES:

```
use ESMF
use MAPL_Mod
use AeroOptPropTableMod

use cudafor
! NOTE: USE renames are used below to prevent name clashes with
!       CUDA copies to the GPU.
use sorad_constants, only: &
      ZK_UV_CONST = >ZK_UV, WK_UV_CONST = >WK_UV, RY_UV_CONST = >RY_UV, &
      XK_IR_CONST = >XK_IR, RY_IR_CONST = >RY_IR,                      &
        COA_CONST = >COA,     CAH_CONST = >CAH
use rad_constants, only: &
       AIB_UV_CONST = >AIB_UV,   AWB_UV_CONST = >AWB_UV,   ARB_UV_CONST = >ARB_UV,  &
       AIG_UV_CONST = >AIG_UV,   AWG_UV_CONST = >AWG_UV,   ARG_UV_CONST = >ARG_UV,  &
      AIB_NIR_CONST = >AIB_NIR, AWB_NIR_CONST = >AWB_NIR, ARB_NIR_CONST = >ARB_NIR, &
      AIA_NIR_CONST = >AIA_NIR, AWA_NIR_CONST = >AWA_NIR, ARA_NIR_CONST = >ARA_NIR, &
      AIG_NIR_CONST = >AIG_NIR, AWG_NIR_CONST = >AWG_NIR, ARG_NIR_CONST = >ARG_NIR, &
         CAIB_CONST = >CAIB,       CAIF_CONST = >CAIF
use soradmod, only: &
      ! Subroutines
      SORAD, &
      ! Device Inputs
      COSZ_DEV, PL_DEV, TA_DEV, WA_DEV, OA_DEV, CWC_DEV, FCLD_DEV, REFF_DEV, &
      RSUVBM_DEV, RSUVDF_DEV, RSIRBM_DEV, RSIRDF_DEV, &
      ! Aerosol inputs
      TAUA_DEV, SSAA_DEV, ASYA_DEV, &
      ! Constants in Global Memory
      COA, CAH, CAIB, CAIF, &
      ! Device Outputs
      FLX_DEV, FLC_DEV, FLXU_DEV, FLCU_DEV, &
      FDIRUV_DEV, FDIFUV_DEV, FDIRPAR_DEV, FDIFPAR_DEV, FDIRIR_DEV, FDIFIR_DEV, &
      FLX_SFC_BAND_DEV, &
      ! Constants
       ZK_UV,  WK_UV,  RY_UV, AIB_UV, AWB_UV, ARB_UV, &
      AIG_UV, AWG_UV, ARG_UV,  XK_IR,  RY_IR, AIB_NIR, &
      AWB_NIR, ARB_NIR, AIA_NIR, AWA_NIR, ARA_NIR, AIG_NIR, &
      AWG_NIR, ARG_NIR, HK_UV, HK_IR
```

```
use soradmod, only: SORAD
use sorad_constants, only : HK_IR_OLD, HK_UV_OLD
use gettau, only: getvistau
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices

!GLOBAL PARAMETERS
INTEGER, PARAMETER :: NB_CHOU_UV   =  5 ! Number of UV bands
INTEGER, PARAMETER :: NB_CHOU_NIR  =  3 ! Number of near-IR bands
INTEGER, PARAMETER :: NB_CHOU      =  NB_CHOU_UV + NB_CHOU_NIR ! Total number of bands
```

34.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer, optional                  :: RC  ! return code
```

DESCRIPTION:
This version uses the MAPL_GenericSetServices. This function sets the Initialize and Finalize services, as well as allocating our instance of a MAPL_MetaComp and putting it in the gridded component (GC). Here we only need to register the Run method with ESMF and register the state variable specifications with MAPL.

STATES:

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
$\overline{\text{PLE}}$	IM	Pa	xyz	Edge	air pressure
$\overline{\text{T}}$	IM	K	xyz	Center	air temperature

Short Name	Type	Units	Dims	Vert Loc	Long name
$\overline{\mathrm{QV}}$	IM	kg kg^{-1}	xyz	Center	specific humidity
$\overline{\mathrm{QL}}$	IM	kg kg^{-1}	xyz	Center	mass fraction of cloud liquid water in air
$\overline{\mathrm{QI}}$	IM	kg kg^{-1}	xyz	Center	mass fraction of cloud ice in air
$\overline{\mathrm{QR}}$	IM	kg kg^{-1}	xyz	Center	mass fraction of rain water in air
$\overline{\mathrm{QS}}$	IM	kg kg^{-1}	xyz	Center	mass fraction of snow in air
$\overline{\mathrm{RL}}$	IM	m	xyz	Center	effective radius of cloud liquid water particles
$\overline{\mathrm{RI}}$	IM	m	xyz	Center	effective radius of cloud ice particles
$\overline{\mathrm{RR}}$	IM	m	xyz	Center	effective radius of rain particles
$\overline{\mathrm{RS}}$	IM	m	xyz	Center	effective radius of snow particles
$\overline{\mathrm{OX}}$	IM	mol mol^{-1}	xyz	Center	odd-oxygen volume mixing ratio
$\overline{\mathrm{FCLD}}$	IM	1	xyz	Center	cloud area fraction
AERO	IM	kg kg^{-1}	xyz	Center	aerosols
ALBVR	IM	1	xy	None	surface albedo for visible beam
ALBVF	IM	1	xy	None	surface albedo for visible diffuse
ALBNR	IM	1	xy	None	surface albedo for near infrared beam
ALBNF	IM	1	xy	None	surface albedo for near infrared diffuse
SWNDSRF	IM	W m^{-2}	xy	None	surface net downward shortwave flux
PREF	IM	Pa	z	Edge	reference air pressure
FSWN	IN	1	xyz	Edge	normalized net downward shortwave flux in air
FSCN	IN	1	xyz	Edge	normalized net downward shortwave flux in air assuming clear sky
FSWUN	IN	1	xyz	Edge	normalized upward shortwave flux in air
FSCUN	IN	1	xyz	Edge	normalized upward shortwave flux in air assuming clear sky
FSWBANDN	IN	1	xy	None	normalized net surface downward shortwave flux per band in air

Short Name	Type	Units	Dims	Vert Loc	Long name
DRUVRN	IN	1	xy	None	normalized surface downwelling ultraviolet beam flux
DFUVRN	IN	1	xy	None	normalized surface downwelling ultraviolet diffuse flux
DRPARN	IN	1	xy	None	normalized surface downwelling par beam flux
DFPARN	IN	1	xy	None	normalized surface downwelling par diffuse flux
DRNIRN	IN	1	xy	None	normalized surface downwelling nearinfrared beam flux
DFNIRN	IN	1	xy	None	normalized surface downwelling nearinfrared diffuse flux
FSWNAN	IN	1	xyz	Edge	normalized net downward shortwave flux in air assuming no aerosol
FSCNAN	IN	1	xyz	Edge	normalized net downward shortwave flux in air assuming clear sky and no aerosol
FSWUNAN	IN	1	xyz	Edge	normalized upward shortwave flux in air assuming no aerosol
FSCUNAN	IN	1	xyz	Edge	normalized upward shortwave flux in air assuming clear sky and no aerosol
FSWBANDNAN	IN	1	xy	None	normalized net surface downward shortwave flux per band in air assuming no aerosol
FSW	EX	W m^{-2}	xyz	Edge	net downward shortwave flux in air
FSC	EX	W m^{-2}	xyz	Edge	net downward shortwave flux in air assuming clear sky
FSWNA	EX	W m^{-2}	xyz	Edge	net downward shortwave flux in air assuming no aerosol
FSCNA	EX	W m^{-2}	xyz	Edge	net downward shortwave flux in air assuming clear sky and no aerosol
FSWD	EX	W m^{-2}	xyz	Edge	downward shortwave flux in air
FSCD	EX	W m^{-2}	xyz	Edge	downward shortwave flux in air assuming clear sky
FSWDNA	EX	W m^{-2}	xyz	Edge	downward shortwave flux in air assuming no aerosol

Short Name	Type	Units	Dims	Vert Loc	Long name
FSCDNA	EX	W m^{-2}	xyz	Edge	downward shortwave flux in air assuming clear sky and no aerosol
FSWU	EX	W m^{-2}	xyz	Edge	upward shortwave flux in air
FSCU	EX	W m^{-2}	xyz	Edge	upward shortwave flux in air assuming clear sky
FSWUNA	EX	W m^{-2}	xyz	Edge	upward shortwave flux in air assuming no aerosol
FSCUNA	EX	W m^{-2}	xyz	Edge	upward shortwave flux in air assuming clear sky and no aerosol
FSWBAND	EX	W m^{-2}	xy	None	net surface downward shortwave flux per band in air
FSWBANDNA	EX	W m^{-2}	xy	None	net surface downward shortwave flux per band in air assuming no aerosol
DRUVRN	EX	1	xy	None	normalized surface downwelling ultraviolet beam flux
DFUVRN	EX	1	xy	None	normalized surface downwelling ultraviolet diffuse flux
DRPARN	EX	1	xy	None	normalized surface downwelling par beam flux
DFPARN	EX	1	xy	None	normalized surface downwelling par diffuse flux
DRNIRN	EX	1	xy	None	normalized surface downwelling nearinfrared beam flux
DFNIRN	EX	1	xy	None	normalized surface downwelling nearinfrared diffuse flux
DRNUVR	EX	W m^{-2}	xy	None	surface downwelling ultraviolet beam normal flux
DRNPAR	EX	W m^{-2}	xy	None	surface downwelling par beam normal flux
DRNNIR	EX	W m^{-2}	xy	None	surface downwelling nearinfrared beam normal flux
DRUVR	EX	W m^{-2}	xy	None	surface downwelling ultraviolet beam flux
DFUVR	EX	W m^{-2}	xy	None	surface downwelling ultraviolet diffuse flux
DRPAR	EX	W m^{-2}	xy	None	surface downwelling par beam flux
DFPAR	EX	W m^{-2}	xy	None	surface downwelling par diffuse flux

Short Name	Type	Units	Dims	Vert Loc	Long name
DRNIR	EX	W m^{-2}	xy	None	surface downwelling nearinfrared beam flux
DFNIR	EX	W m^{-2}	xy	None	surface downwelling nearinfrared diffuse flux
FCLD	EX	1	xyz	Center	cloud area fraction
CLDLO	EX	1	xy	None	cloud area fraction for low clouds
CLDMD	EX	1	xy	None	cloud area fraction for middle clouds
CLDHI	EX	1	xy	None	cloud area fraction for high clouds
CLDTT	EX	1	xy	None	total cloud area fraction
TAULO	EX	1	xy	None	in cloud optical thickness of low clouds
TAUMD	EX	1	xy	None	in cloud optical thickness of middle clouds
TAUHI	EX	1	xy	None	in cloud optical thickness of high clouds(EXPORT)
TAUTT	EX	1	xy	None	in cloud optical thickness of all clouds
TAUCLI	EX	1	xyz	Center	in cloud optical thickness for ice clouds
TAUCLW	EX	1	xyz	Center	in cloud optical thickness for liquid clouds
TAUCLR	EX	1	xyz	Center	in cloud optical thickness for falling rain
TAUCLS	EX	1	xyz	Center	in cloud optical thickness for falling snow
RSCS	EX	W m^{-2}	xy	None	surface net downward shortwave flux assuming clear sky
RSRS	EX	W m^{-2}	xy	None	surface net downward shortwave flux
RSCSNA	EX	W m^{-2}	xy	None	surface net downward shortwave flux assuming clear sky and no aerosol
RSRSNA	EX	W m^{-2}	xy	None	surface net downward shortwave flux assuming no aerosol
SLRSF	EX	W m^{-2}	xy	None	surface incoming shortwave flux
SLRSFC	EX	W m^{-2}	xy	None	surface incoming shortwave flux assuming clear sky
SLRSFNA	EX	W m^{-2}	xy	None	surface incoming shortwave flux assuming clean sky

Short Name	Type	Units	Dims	Vert Loc	Long name
SLRSFCNA	EX	W m^{-2}	xy	None	surface incoming shortwave flux assuming clear clean sky
SLRSUF	EX	W m^{-2}	xy	None	surface outgoing shortwave flux
SLRSUFC	EX	W m^{-2}	xy	None	surface outgoing shortwave flux assuming clear sky
SLRSUFNA	EX	W m^{-2}	xy	None	surface outgoing shortwave flux assuming clean sky
SLRSUFCNA	EX	W m^{-2}	xy	None	surface outgoing shortwave flux assuming clear clean sky
OSR	EX	W m^{-2}	xy	None	toa outgoing shortwave flux
OSRCLR	EX	W m^{-2}	xy	None	toa outgoing shortwave flux assuming clear sky
OSRNA	EX	W m^{-2}	xy	None	toa outgoing shortwave flux no aerosol
OSRCNA	EX	W m^{-2}	xy	None	toa outgoing shortwave flux no aerosol clear sky
RSR	EX	W m^{-2}	xy	None	toa net downward shortwave flux
RSC	EX	W m^{-2}	xy	None	toa net downward shortwave flux assuming clear sky
RSRNA	EX	W m^{-2}	xy	None	toa net downward shortwave flux assuming no aerosol
RSCNA	EX	W m^{-2}	xy	None	toa net downward shortwave flux assuming clear sky and no aerosol
SLRTP	EX	W m^{-2}	xy	None	toa incoming shortwave flux
ALBEDO	EX	1	xy	None	surface albedo
ALBVR	EX	1	xy	None	surface albedo for visible beam
ALBVF	EX	1	xy	None	surface albedo for visible diffuse
ALBNR	EX	1	xy	None	surface albedo for near infrared beam
ALBNF	EX	1	xy	None	surface albedo for near infrared diffuse
COSZ	EX	1	xy	None	cosine of the solar zenith angle
MCOSZ	EX	1	xy	None	mean cosine of the solar zenith angle
TAUA1	EX	1	xyz	Center	aerosol optical thickness in 0.225-0.285 band

Short Name	Type	Units	Dims	Vert Loc	Long name
TAUA2	EX	1	xyz	Center	aerosol optical thickness in 0.175-0.225 0.285-0.300 band
TAUA3	EX	1	xyz	Center	aerosol optical thickness in 0.300-0.325 band
TAUA4	EX	1	xyz	Center	aerosol optical thickness in 0.325-0.4 band
TAUA5	EX	1	xyz	Center	aerosol optical thickness in
TAUA6	EX	1	xyz	Center	aerosol optical thickness in 0.690-1.220 band
TAUA7	EX	1	xyz	Center	aerosol optical thickness in 1.220-2.270 band
TAUA8	EX	1	xyz	Center	aerosol optical thickness in 2.270-3.850 band
TAUDU	EX	1	xyz	Center	dust optical thickness in 0.4-0.690 band
TAUSS	EX	1	xyz	Center	salt optical thickness in 0.4-0.690 band
TAUSO	EX	1	xyz	Center	sulfate optical thickness in 0.4-0.690 band
TAUBC	EX	1	xyz	Center	black carbon optical thickness in 0.4-0.690 band
TAUOC	EX	1	xyz	Center	organic carbon optical thickness in 0.4-0.690 band
TTAUDU	EX	1	xy	None	total dust optical thickness in 0.4-0.690 band
TTAUSS	EX	1	xy	None	total salt optical thickness in 0.4-0.690 band
TTAUSO	EX	1	xy	None	total sulfate optical thickness in 0.4-0.690 band
TTAUBC	EX	1	xy	None	total black carbon optical thickness in 0.4-0.690 band
TTAUOC	EX	1	xy	None	total organic carbon optical thickness in 0.4-0.690 band
TDUST	EX	kg m^{-2}	xy	None	total dust aerosol loading
TSALT	EX	kg m^{-2}	xy	None	total sea salt aerosol loading
TSO4	EX	kg m^{-2}	xy	None	total sulfate aerosol loading
TBC	EX	kg m^{-2}	xy	None	total black carbon aerosol loading
TOC	EX	kg m^{-2}	xy	None	total organic carbon aerosol loading
CLDTMP	EX	K	xy	None	cloud top temperature
CLDPRS	EX	Pa	xy	None	cloud top pressure

34.2 RUN – Run method for the SOLAR component

INTERFACE:

```
subroutine RUN ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
  type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
  type(ESMF_State),    intent(inout) :: IMPORT ! Import state
  type(ESMF_State),    intent(inout) :: EXPORT ! Export state
  type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
  integer, optional,   intent(  out) :: RC     ! Error code:
/*
```

DESCRIPTION:
Each time the Run method is called it fills all Exports for which an allocated pointer is available. Exports are filled from the normalized fluxes kept in the Internal state and the position of the Sun for the current interval in the Clock. If MAPL's RunAlarm is ringing, it also refreshes the normalized fluxes kept in the internal state by doing a full transfer calculation valid for solar positions over a "future interval" extebnding to the next anticipated ringing of the RunAlarm. Whether this is done before or after the Exports are updated and the excat definition of the "future interval" is controlled by a flag in the configuration.

A simple load balancing scheme is used that evens work between antipodal processors.

*/ BUGS:

- Deciding on the correct behavior for intermitent calls can be tricky.
- Load-balancing communication needs to be upgraded to most up-to-date ESMF machine model.

35 Module GEOS_SuperdynGridCompMod – A Module to combine Dynamics and Gravity-Wave-Drag Gridded Components

USES:

```
use ESMF
use MAPL_Mod

use FVdycore_GridCompMod,     only :    FV_SetServices  = > SetServices
use FVdycoreCubed_GridComp,   only :   FV3_SetServices  = > SetServices
use ARIESg3_GridCompMod,      only : ARIES_SetServices  = > SetServices
use GEOS_DatmoDynGridCompMod, only : DATMO_SetServices  = > SetServices
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

DESCRIPTION:
This gridded component (GC) combines the Dynamics GC and the Gravity Wave Drag GC into a new composite SuperDyn GC.

Import Couplings:
The Import Couplings of the SuperDyn GC are the tendencies of the atmospheric state variables U,V,T,PE (due to external diabatic forcing) in addition to a collection of "Friendly" tracers for advection. The Friendlies will be searched for moisture for use in virtual effects in both the Dynamics and Gravity Wave Drag parameterization. If no moisture is found, the SyperDyn will be run dry.

```
DUDT .... U-Wind                      Tendency (m/s)
DVDT .... V-Wind                      Tendency (m/s)
DPEDT ... Edge-Pressure               Tendency (Pa/s)
DTDT .... Mass-Weighted Temperature Tendency (Pa K/s)
TRACER .. Friendly Tracers                     (unknown)
```

Run Method:
The run method first calls the Gravity Wave Drag parameterization. The tendencies of the atmospheric state variables created by the GWD are then ADDED to the SuperDyn Import Couplings (i.e., state variable tendencies due to external diabatic forcing), which are then used to force the Dynamics GC.

Export Couplings:
The Export Couplings of the SuperDyn GC are the union of the Export Couplings of the individual GCs. It should be noted that the SuperDyn GC controls the GEOS Topo Utility and produces Topo Variables based on the Grid defined by the DYN GC. The Topo Variables are computed during the SuperDyn Initialize method, and are part of the SuperDyn Export Couplings. ESMF utilities may be used to regrid these Topo Variables to other components with differing Grids.

35.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT)      :: GC  ! gridded component
integer, optional,   intent(  OUT)      :: RC  ! return code
```

DESCRIPTION:
The SetServices for the SuperDyn GC needs to register its Initialize, Run, and Finalize methods. In addition, we need to create the children GCs (DYN and GWD) and run their respective SetServices. *RESOURCES:*

Name	Description	Units	Default
'DYCORE:'			"FV"

STATES:

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
DUDT	IM	m s^{-2}	xyz	Center	eastward wind tendency
DVDT	IM	m s^{-2}	xyz	Center	northward wind tendency
DTDT	IM	Pa K s^{-1}	xyz	Center	delta-p weighted temperature tendency
U	EX				
V	EX				
T	EX				
S	EX				
TH	EX				
PLE	EX				
PL	EX				
ZLE	EX				
PREF	EX				
AK	EX				
BK	EX				
PLK	EX				
PS	EX				

Short Name	Type Units	Dims	Vert Loc	Long name
TA	EX			
QA	EX			
SPEED	EX			
DZ	EX			
TROPP_BLENDED	EX			
PV	EX			
TV	EX			
OMEGA	EX			
EPV	EX			
PEANA	EX			
DTHVDTANAINT	EX			
PEPHY	EX			
DTHVDTPHYINT	EX			
DQVDTANAINT	EX			
DQLDTANAINT	EX			
DQIDTANAINT	EX			
DOXDTANAINT	EX			
AREA	EX			
U_DGRID	EX			
V_DGRID	EX			
PT	EX			
PE	EX			

35.2 Initialize – Initialize method for the composite SuperDyn Gridded Component

INTERFACE:

```
subroutine Initialize ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code
```

DESCRIPTION:

The Initialize method of the SuperDyn Composite Gridded Component first calls the Initialize method of the child Dynamics. The Dynamics Initialize method will create the ESMF

GRID, which will then be used to set the GRID associated with the SuperDyn Composite Component itself. It should be noted that the SuperDyn Initialize method also invokes the GEOS Topo Utility which creates all topography related quantities.

35.3 Run – Run method for the composite SuperDyn Gridded Component

INTERFACE:

```
subroutine Run ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code
```

DESCRIPTION:
The run method first calls the Gravity Wave Drag parameterization. The tendencies of the atmospheric state variables created by the GWD are then ADDED to the SuperDyn Import Couplings (i.e., state variable tendencies due to external diabatic forcing), which are then used to force the Dynamics GC.

36 Module GEOS_Surface — A composite component for the surface components.

DESCRIPTION:
`GEOS_Surface` is a light-weight gridded component that implements the interface to the tiled surface components. The surface computational components (LAND, LAKE, OCEAN, LANDICE) are its children. All of `GEOS_Surface`'s imports and exports are in the atmospheric model's grid. In `GEOS_Surface` these are transformed to the exchange grid, and the relevant portions of the exchange grid are passed to each of the children. The children's results are them replaced in the full exchange grid and transformed back to the atmospheric grid.
`GEOS_Surface` has two run stages, as do its children. These are meant to interface with the two stages of `GEOS_Turbulence`. During the first run stage, the children all produce surface exchange coefficients, and during the second, they update the surface state and produce final values of the fluxes.
`GEOS_Surface` keeps a Private Internal State called 'SURF_state' in the component object. In this state it saves the tranforms between the atmospheric grid and each of the children's exchange grids. This should be done more elegantly once ESMF has exchange grid support.

It also has a Internal State that is used to communicate between the two run methods. These internal states do not need to be saved in restarts.
The four children of `GEOS_Surface` are given the names: 'LAKE', which treats inland freshwater bodies; 'LANDICE', which treats permanent glaciers; 'LAND', which treats all other land surface types, both bare and vegetated, as well as vegetated wetlands not considered freshwater bodies; and 'SALTWATER', which performs the surface calculations for all ocean areas. All four operate in lists of tiles that are nonoverlapping subsets of the exhange grid, and their union—the full exchange grid—tiles the entire sphere.
By default MAPL_Generic tries to resolve Imports and Exports among the children; but the children of `GEOS_Surface` do not talk directly to each other, and all communication between them would need to be performed by `GEOS_Surface` manipulating their Import and Export states. *USES:*

```
use ESMF
use MAPL_Mod
use GEOS_UtilsMod

use GEOS_LakeGridCompMod,      only : LakeSetServices      = > SetServices
use GEOS_LandiceGridCompMod,   only : LandiceSetServices   = > SetServices
use GEOS_LandGridCompMod,      only : LandSetServices      = > SetServices
use GEOS_CICEGridCompMod,      only : OceanSetServices     = > SetServices
use GEOS_SaltwaterGridCompMod, only : OceanSetServices     = > SetServices
use m_mpif90, only: MP_INTEGER, MP_REAL, MP_STATUS_SIZE


real(kind = ESMF_KIND_R8), parameter :: pi  =  3.14159265358979323846
real(kind = ESMF_KIND_R8), parameter :: rad_to_deg  =  180.0/pi  ! degree-radian conversion

type( ESMF_VM ) :: VMG
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

36.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer,             intent(  OUT) :: RC  ! return code
```

DESCRIPTION:
This version uses the GEOS_GenericSetServices, which in addition to setting default IRF methods, also allocates our instance of a generic state and puts it in the gridded component (GC). Here we override the Initialize and Run methods. The Run method is a two-stage method that implemets the interaction between the 2-stage children representing the various surface types and the 2-stage turbulence run methods.

Note that, in addition to its explicit exports, the entire internal state, which is used to communicate between the two run stages, is exported using the "friendly-to-self" mechanism.

Imports are read-only quantities computed by other gridded components.

Note that the turbulence fluxes appearing in the import state are the values computed by the first run stage of turbulence using fixed surface conditions. The Export versions of these fluxes are the final values actually used in the surface budgets. The same applies to some of the radiative fluxes, for which the values exported here are those actually used in the budget. *STATES:*

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
PS	IM	Pa	xy	None	surface pressure
TA	IM	K	xy	None	surface air temperature
QA	IM	kg kg^{-1}	xy	None	surface air specific humidity
SPEED	IM	m s^{-1}	xy	None	surface wind speed
UA	IM	m s^{-1}	xy	None	eastward wind bottom level
VA	IM	m s^{-1}	xy	None	northward wind bottom level
DZ	IM	m	xy	None	surface layer height
PHIS	IM	m^2 s^{-2}	xy	None	surface
SH	IM	W m^{-2}	xy	None	sensible heat flux
TAUX	IM	N m^{-2}	xy	None	eastward surface stress on air
TAUY	IM	N m^{-2}	xy	None	northward surface stress on air
EVAP	IM	kg m^{-2} s^{-1}	xy	None	evaporation
DEWL	IM	kg m^{-2} s^{-1}	xy	None	dewfall
FRSL	IM	kg m^{-2} s^{-1}	xy	None	frostfall
DSH	IM	m s^{-1}	xy	None	derivative of sensible heat wrt dry static energy
DFU	IM	N s m^{-3}	xy	None	derivative of eastward surface stress wrt Us

Short Name	Type	Units	Dims	Vert Loc	Long name
DFV	IM	N s m^{-3}	xy	None	derivative of northward surface stress wrt Us
DEVAP	IM	kg m^{-2} s^{-1}	xy	None	derivative of evaporation wrt QS
DDEWL	IM	kg m^{-2} s^{-1}	xy	None	derivative of dewfall wrt QS
DFRSL	IM	kg m^{-2} s^{-1}	xy	None	derivative of frostfall wrt QS
PCU	IM	kg m^{-2} s^{-1}	xy	None	liquid water convective precipitation
PLS	IM	kg m^{-2} s^{-1}	xy	None	liquid water large scale precipitation
SNO	IM	kg m^{-2} s^{-1}	xy	None	snowfall
DRPARN	IM	1	xy	None	normalized surface downwelling par beam flux
DFPARN	IM	1	xy	None	normalized surface downwelling par diffuse flux
DRNIRN	IM	1	xy	None	normalized surface downwelling nir beam flux
DFNIRN	IM	1	xy	None	normalized surface downwelling nir diffuse flux
DRUVRN	IM	1	xy	None	normalized surface downwelling uvr beam flux
DFUVRN	IM	1	xy	None	normalized surface downwelling uvr diffuse flux
LWDNSRF	IM	W m^{-2}	xy	None	surface downwelling longwave flux
ALW	IM	W m^{-2}	xy	None	linearization of surface upwelling longwave flux
BLW	IM	W m^{-2} K^{-1}	xy	None	linearization of surface upwelling longwave flux
CO2SC	IM	$1e^{-6}$	xy	None	CO2
FSWBAND	IM	W m^{-2}	xy	None	net surface downward shortwave flux per band in air
FSWBANDNA	IM	W m^{-2}	xy	None	net surface downward shortwave flux per band in air assuming no aerosol
AERO_DP	IM	kg m^{-2} s^{-1}	xy	None	aerosol deposition
DTSDT	IM	K s^{-1}	xy	None	skin temperature analysis tendency
ALBVR	EX	1	xy	None	surface albedo for visible beam
ALBVF	EX	1	xy	None	surface albedo for visible diffuse

Short Name	Type	Units	Dims	Vert Loc	Long name
ALBNR	EX	1	xy	None	surface albedo for nearinfrared beam
ALBNF	EX	1	xy	None	surface albedo for nearinfraed diffuse
EMIS	EX	1	xy	None	surface emissivity
Z0	EX	m	xy	None	surface roughness
Z0H	EX	m	xy	None	surface roughness for heat
RI	EX	1	xy	None	surface bulk richardson number
RE	EX	1	xy	None	surface reynolds number
FRACI	EX	1	xy	None	ice covered fraction of tile
QDWL	EX	kg kg^{-1}	xy	None	surface liquid condensate
QFRL	EX	kg kg^{-1}	xy	None	surface ice condensate
SHAT	EX	m^2 s^{-2}	xy	None	effective surface dry static energy
DELUS	EX	m s^{-1}	xy	None	change of surface eastward velocity
DELVS	EX	m s^{-1}	xy	None	change of surface northward velocity
DELSS	EX	m^2 s^{-2}	xy	None	change of surface dry static energy
DELTS	EX	K	xy	None	change of surface skin temperature
DELQS	EX	kg kg^{-1}	xy	None	change of surface specific humidity
DLQLL	EX	kg kg^{-1}	xy	None	change of surface liquid condensate
DLQIL	EX	kg kg^{-1}	xy	None	change of surface frozen condensate
FRLAND	EX	1	xy	None	fraction of land
FRLANDICE	EX	1	xy	None	fraction of land ice
FRLAKE	EX	1	xy	None	fraction of lake
FROCEAN	EX	1	xy	None	fraction of ocean
USTAR	EX	m s^{-1}	xy	None	surface velocity scale
TSTAR	EX	K	xy	None	surface temperature scale
QSTAR	EX	kg kg^{-1}	xy	None	surface moisture scale
BSTAR	EX	m s^{-2}	xy	None	surface bouyancy scale
TSOIL1	EX	K	xy	None	soil temperatures layer 1
TSOIL2	EX	K	xy	None	soil temperatures layer 2
TSOIL3	EX	K	xy	None	soil temperatures layer 3
TSOIL4	EX	K	xy	None	soil temperatures layer 4
TSOIL5	EX	K	xy	None	soil temperatures layer 5
TSOIL6	EX	K	xy	None	soil temperatures layer 6

Short Name	Type	Units	Dims	Vert Loc	Long name
ASNOW	EX	1	xy	None	fractional area of land snowcover
SHSNOW	EX	W m^{-2}	xy	None	downward heat flux into snow
AVETSNOW	EX	K	xy	None	averaged snow temperature
TPSNOW	EX	K	xy	None	surface temperature of snow
TPSAT	EX	K	xy	None	surface temperature of saturated zone
TPUNST	EX	K	xy	None	surface temperature of unsaturated zone
TPWLT	EX	K	xy	None	surface temperature of wilted zone
TPSURF	EX	K	xy	None	surface temperature of land incl snow
FRSAT	EX	1	xy	None	fractional area of saturated zone
FRUST	EX	1	xy	None	fractional area of unsaturated zone
FRWLT	EX	1	xy	None	fractional area of wilting zone
SNOMAS	EX	kg m^{-2}	xy	None	snow mass
WET1	EX	1	xy	None	surface soil wetness
WET2	EX	1	xy	None	root zone soil wetness
WET3	EX	1	xy	None	ave prof soil moisture
WCSF	EX	m^{-3} m^{-3}	xy	None	water surface layer
WCRZ	EX	m^{-3} m^{-3}	xy	None	water root zone
WCPR	EX	m^{-3} m^{-3}	xy	None	water profile
LAI	EX	1	xy	None	leaf area index
GRN	EX	1	xy	None	greeness fraction
Z2CH	EX	m	xy	None	canopy height
ROOTL	EX	m m^{-3}	xy	None	root length
SH	EX	W m^{-2}	xy	None	sensible heat flux from turbulence
TAUX	EX	N m^{-2}	xy	None	eastward surface stress
TAUY	EX	N m^{-2}	xy	None	northward surface stress
EVAP	EX	kg m^{-2} s^{-1}	xy	None	evaporation from turbulence
U10M	EX	m s^{-1}	xy	None	10-meter eastward wind
V10M	EX	m s^{-1}	xy	None	10-meter northward wind
U10N	EX	m s^{-1}	xy	None	equivalent neutral 10-meter eastward wind
V10N	EX	m s^{-1}	xy	None	equivalent neutral 10-meter northward wind
U50M	EX	m s^{-1}	xy	None	50-meter eastward wind
V50M	EX	m s^{-1}	xy	None	50-meter northward wind

Short Name	Type	Units	Dims	Vert Loc	Long name
T10M	EX	K	xy	None	10-meter air temperature
Q10M	EX	kg kg^{-1}	xy	None	10-meter specific humidity
U2M	EX	m s^{-1}	xy	None	2-meter eastward wind
V2M	EX	m s^{-1}	xy	None	2-meter northward wind
T2M	EX	K	xy	None	2-meter air temperature
Q2M	EX	kg kg^{-1}	xy	None	2-meter specific humidity
TA	EX	K	xy	None	surface air temperature
QA	EX	kg kg^{-1}	xy	None	surface air specific humidity
UA	EX	m s^{-1}	xy	None	surface eastward wind
VA	EX	m s^{-1}	xy	None	surface northward wind
GUST	EX	m s^{-1}	xy	None	gustiness
VENT	EX	m s^{-1}	xy	None	surface ventilation velocity
LWI	EX	1	xy	None	land(1) water(0) ice(2) flag
SNOWDP	EX	m	xy	None	snow depth
TAUXW	EX	N m^{-2}	xy	None	eastward stress over water
TAUYW	EX	N m^{-2}	xy	None	northward stress over water
TAUXI	EX	N m^{-2}	xy	None	eastward stress over ice
TAUYI	EX	N m^{-2}	xy	None	northward stress over ice
SHWTR	EX	W m^{-2}	xy	None	open water upward sensible heat flux
SHICE	EX	W m^{-2}	xy	None	sea ice upward sensible heat flux
HLATWTR	EX	W m^{-2}	xy	None	open water latent energy flux
HLATICE	EX	W m^{-2}	xy	None	sea ice latent energy flux
LWNDWTR	EX	W m^{-2}	xy	None	open water net downward longwave flux
LWNDICE	EX	W m^{-2}	xy	None	sea ice net downward longwave flux
SWNDWTR	EX	W m^{-2}	xy	None	open water net downward shortwave flux
SWNDICE	EX	W m^{-2}	xy	None	sea ice net downward shortwave flux
SNOWOCN	EX	kg m^{-2} s^{-1}	xy	None	ocean snowfall
RAINOCN	EX	kg m^{-2} s^{-1}	xy	None	ocean rainfall
EVAPOUT	EX	kg m^{-2} s^{-1}	xy	None	evaporation
SUBLIM	EX	kg m^{-2} s^{-1}	xy	None	sublimation
SHOUT	EX	W m^{-2}	xy	None	upward sensible heat flux
LST	EX	K	xy	None	land surface skin temperature
CDCR2	EX	kg m^{-2}	xy	None	max water content
DISCHARGE	EX	kg m^{-2} s^{-1}	xy	None	river discharge at ocean points
DISCHARGE_IN	EX	kg m^{-2} s^{-1}	xy	None	river discharge at ocean points

Short Name	Type	Units	Dims	Vert Loc	Long name
DISCHARGE_OUT	EX	kg m^{-2} s^{-1}	xy	None	river discharge at ocean points
RUNOFF	EX	kg m^{-2} s^{-1}	xy	None	runoff flux
DRAINAGE	EX	kg m^{-2} s^{-1}	xy	None	river drainage at ocean points
EVPINT	EX	W m^{-2}	xy	None	interception loss energy flux
EVPSOI	EX	W m^{-2}	xy	None	baresoil evap energy flux
EVPVEG	EX	W m^{-2}	xy	None	transpiration energy flux
EVPICE	EX	W m^{-2}	xy	None	snow ice evaporation energy flux
EVPSNO	EX	W m^{-2}	xy	None	snowpack evaporation energy flux
WAT10CM	EX	kg m^{-2}	xy	None	soil
WATSOI	EX	kg m^{-2}	xy	None	total
ICESOI	EX	kg m^{-2}	xy	None	soil
BASEFLOW	EX	kg m^{-2} s^{-1}	xy	None	baseflow flux
RUNSURF	EX	kg m^{-2} s^{-1}	xy	None	surface runoff flux
EVLAND	EX	kg m^{-2} s^{-1}	xy	None	Evaporation land
LHLAND	EX	W m^{-2}	xy	None	Latent heat flux land
SHLAND	EX	W m^{-2}	xy	None	Sensible heat flux land
SWLAND	EX	W m^{-2}	xy	None	Net shortwave land
SWDOWNLAND	EX	W m^{-2}	xy	None	Incident shortwave land
LWLAND	EX	W m^{-2}	xy	None	Net longwave land
GHLAND	EX	W m^{-2}	xy	None	Ground heating land
GHSNOW	EX	W m^{-2}	xy	None	Ground heating snow
GHTSKIN	EX	W m^{-2}	xy	None	Ground heating for skin temp
SMLAND	EX	kg m^{-2} s^{-1}	xy	None	Snowmelt flux land
QINFIL	EX	kg m^{-2} s^{-1}	xy	None	Soil water infiltration rate
TWLAND	EX	kg m^{-2}	xy	None	Avail water storage land
TELAND	EX	J m^{-2}	xy	None	Total energy storage land
TSLAND	EX	kg m^{-2}	xy	None	Total snow storage land
DWLAND	EX	kg m^{-2} s^{-1}	xy	None	rate of change of total land water
DHLAND	EX	W m^{-2}	xy	None	rate of change of total land energy
SPLAND	EX	W m^{-2}	xy	None	rate of spurious land energy source
SPWATR	EX	kg m^{-2} s^{-1}	xy	None	rate of spurious land water source
SPSNOW	EX	W m^{-2}	xy	None	rate of spurious snow energy
SMELT	EX	kg m^{-2} s^{-1}	xy	None	snowmelt flux
HLWUP	EX	W m^{-2}	xy	None	surface outgoing longwave flux
LWNDSRF	EX	W m^{-2}	xy	None	surface net downward longwave flux

Short Name	Type	Units	Dims	Vert Loc	Long name
SWNDSRF	EX	W m^{-2}	xy	None	surface net downward shortwave flux
LHFX	EX	W m^{-2}	xy	None	total latent energy flux
ACCUM	EX	kg m^{-2} s^{-1}	xy	None	net ice accumulation rate
ITY	EX	1	xy	None	vegetation type
NITY	EX	1	xy	None	NCEP vegetation type
PCU	EX	kg m^{-2} s^{-1}	xy	None	liquid water convective precipitation
PLS	EX	kg m^{-2} s^{-1}	xy	None	liquid water large scale precipitation
PRECTOT	EX	kg m^{-2} s^{-1}	xy	None	total precipitation
SNO	EX	kg m^{-2} s^{-1}	xy	None	snowfall
TSKINW	EX	K	xy	None	open water skin temperature
HICE	EX	m	xy	None	grid cell mean ice thickness
HSNO	EX	m	xy	None	grid cell mean snow thickness
FRZMLT	EX	W m^{-2}	xy	None	freezing melting potential
TSKINWCICE	EX	K	xy	None	CICE water skin temperature
ISTSFC	EX	C	xy	None	snow or ice surface temperature
SSKINW	EX	psu	xy	None	sea skin layer salinity
MELTT	EX	m s^{-1}	xy	None	top ice melt
MELTB	EX	m s^{-1}	xy	None	basal ice melt
MELTL	EX	m s^{-1}	xy	None	lateral ice melt
MELTS	EX	m s^{-1}	xy	None	snow melt
FRAZIL	EX	m s^{-1}	xy	None	frazil ice growth
CONGEL	EX	m s^{-1}	xy	None	congelation ice growth
SNOICE	EX	m s^{-1}	xy	None	snow-ice formation
DAIDTT	EX	% day^{-1}	xy	None	ice area tendency dueto thermodynamics
DVIDTT	EX	cm day^{-1}	xy	None	ice volume tendency dueto thermodynamics
DAIDTD	EX	% day^{-1}	xy	None	ice area tendency dueto dynamics
DVIDTD	EX	cm day^{-1}	xy	None	ice volume tendency dueto dynamics
FBOT	EX	W m^{-2}	xy	None	net downward heat flux from ice to ocean
HFLUX	EX	W m^{-2}	xy	None	heat flux bw saltwater ocean
WATERFLUX	EX	kg m^{-2} s^{-1}	xy	None	FRESHWATER flux bw saltwater ocean
SALTFLUX	EX	kg m^{-2} s^{-1}	xy	None	salt flux bw saltwater ocean
FSWTHRU	EX	W m^{-2}	xy	None	SW flux thru ice to ocean
FSWABS	EX	W m^{-2}	xy	None	SW flux absorbed by skin layer

Short Name	Type	Units	Dims	Vert Loc	Long name
USTARI	EX	$m\ s^{-1}$	xy	None	ice ocean friction velocity
FHOCN	EX	$W\ m^{-2}$	xy	None	actual ocean ice flux
WESNN1	EX	$kg\ m^{-2}$	xy	None	snow mass layer 1
WESNN2	EX	$kg\ m^{-2}$	xy	None	snow mass layer 2
WESNN3	EX	$kg\ m^{-2}$	xy	None	snow mass layer 3
CAPAC	EX	$kg\ m^{-2}$	xy	None	interception reservoir capac
T2MDEW	EX	K	xy	None	dew point temperature at 2 m
T2MWET	EX	K	xy	None	wet bulb temperature at 2 m
RH2M	EX	xy	None	near-surface relative humidity	
UU10M	EX	$m\ s^{-1}$	xy	None	near-surface wind speed
DCOOL	EX	m	xy	None	depth of cool layer
DWARM	EX	m	xy	None	depth at base of warm layer
TDROP	EX	K	xy	None	temperature drop across cool layer
QCOOL	EX	$W\ m^{-2}$	xy	None	net cooling in cool layer
SWCOOL	EX	$W\ m^{-2}$	xy	None	solar heating in cool layer
USTARW	EX	$m\ s^{-1}$	xy	None	ustar over water layer
TBAR	EX	K	xy	None	mean temperature of interface layer
LCOOL	EX	1	xy	None	Saunders parameter
BCOOL	EX	$m^2\ s^{-3}$	xy	None	bouyancy generation in cool layer
TDEL	EX	K	xy	None	temperature at base of cool layer
TS_FOUND	EX	K	xy	None	foundation temperature for interface layer
QWARM	EX	$W\ m^{-2}$	xy	None	net heating in warm layer
SWWARM	EX	$W\ m^{-2}$	xy	None	solar heating in warm layer
LANGM	EX	1	xy	None	Langmuir number
PHIW	EX	1	xy	None	Similarity function in warm layer
QS	IN	$kg\ kg^{-1}$	xy	None	surface specific humidity
TS	IN	K	xy	None	surface skin temperature
CT	IN	$kg\ m^{-2}\ s^{-1}$	xy	None	surface exchange coefficient for heat
CQ	IN	$kg\ m^{-2}\ s^{-1}$	xy	None	surface exchange coefficient for moisture
CM	IN	$kg\ m^{-2}\ s^{-1}$	xy	None	surface exchange coefficient for momentum
CN	IN	1	xy	None	surface neutral drag coefficient

Short Name	Type	Units	Dims	Vert Loc	Long name
THAT	IN	K	xy	None	effective surface skin temperature
QHAT	IN	kg kg^{-1}	xy	None	effective surface specific humidity
UHAT	IN	m s^{-1}	xy	None	effective surface eastward velocity
VHAT	IN	m s^{-1}	xy	None	effective surface northward velocity
RHOS	IN	kg m^{-3}	xy	None	air density at surface
D0	IN	m	xy	None	zero plane displacement height

36.2 RUN1 – First stage Run method for the Surface component

INTERFACE:

```
subroutine RUN1 ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code:
```

DESCRIPTION:
Interfaces to the children RUN1 methods, which compute the surface exchange coefficients. In addition to exchange coefficients for heat, moisture, and momentum, it also computes effective surface values of the diffused quantities on the atmospheric grid. These are exchange-coefficient-weighted averages of the tile values within an atmospheric grid box.

36.3 RUN2 – Second Run method for the Surface component

INTERFACE:

```
subroutine RUN2 ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code:
```

DESCRIPTION:

37 Module GEOS_Turbulence — An GEOS generic atmospheric turbulence component

USES:

```
use ESMF
use GEOS_Mod
use MAPL_Mod

use cudafor
use LockEntrain, only: &
      ! Subroutines
      ENTRAIN, &
      ! Working Arrays
      ZFULL_DEV, TV_DEV, PV_DEV, RDZ_DEV, &
      DMI_DEV, PFULL_DEV,        &
      ! Inputs - Prelims
      T_DEV, QV_DEV, PHALF_DEV, TH_DEV,       &
      QLCN_DEV, QLLS_DEV, QICN_DEV, QILS_DEV, &
      ! Inputs - Louis
      U_DEV, V_DEV, ZPBL_DEV, &
      ! Inputs - Lock
      TDTLW_IN_DEV, U_STAR_DEV, B_STAR_DEV, FRLAND_DEV, &
      ! Inputs - Postlock
      CT_DEV, CQ_DEV, CU_DEV, &
      ! Inputs - Beljaars
      VARFLT_DEV, &
      ! Outputs - Prelims
      ZHALF_DEV, &
      ! Outputs - Louis
      DIFF_M_DEV, DIFF_T_DEV, &
      RI_DEV, DU_DEV,         &
      ! Outputs - Lock
      K_M_ENTR_DEV, K_T_ENTR_DEV,     &
      K_SFC_DIAG_DEV, K_RAD_DIAG_DEV, &
      ZCLOUD_DEV, ZRADML_DEV, &
```

```
        ZRADBASE_DEV, ZSML_DEV, &
        ! Outputs - Postlock
        AKQ_DEV, AKS_DEV, AKV_DEV, &
        BKQ_DEV, BKS_DEV, BKV_DEV, &
        CKQ_DEV, CKS_DEV, CKV_DEV, &
                          EKV_DEV, &
        ! Outputs - Beljaars
        FKV_DEV, &
        ! Outputs - Decomp
        DKQ_DEV, DKS_DEV, DKV_DEV, &
        ! Diagnostics - Louis
        ALH_DIAG_DEV, KMLS_DIAG_DEV, KHLS_DIAG_DEV, &
        ! Diagnostics - Lock
        ZCLDTOP_DIAG_DEV, WENTR_SFC_DIAG_DEV, WENTR_RAD_DIAG_DEV, &
        DEL_BUOY_DIAG_DEV, VSFC_DIAG_DEV, VRAD_DIAG_DEV,          &
        KENTRAD_DIAG_DEV, VBRV_DIAG_DEV, WENTR_BRV_DIAG_DEV,      &
        DSIEMS_DIAG_DEV, CHIS_DIAG_DEV, DELSINV_DIAG_DEV,         &
        SLMIXTURE_DIAG_DEV, CLDRADF_DIAG_DEV, RADRCODE_DIAG_DEV,  &
        ! Diagnostics - Postlock
        AKQODT_DIAG_DEV, AKSODT_DIAG_DEV, AKVODT_DIAG_DEV, &
        CKQODT_DIAG_DEV, CKSODT_DIAG_DEV, CKVODT_DIAG_DEV, &
        PPBL_DIAG_DEV,   TCZPBL_DIAG_DEV, &
        KPBL_DIAG_DEV,   &
        ZPBL2_DIAG_DEV, ZPBL10p_DIAG_DEV, ZPBLHTKE_DIAG_DEV, &
        ZPBLRI_DIAG_DEV, ZPBLTHV_DIAG_DEV, ZPBLRI2_DIAG_DEV, &
        ! Constants from MAPL_GetResource
        LOUIS_CONST, MINSHEAR_CONST, MINTHICK_CONST, AKHMMAX_CONST,     &
        LAMBDAM_CONST, LAMBDAM2_CONST, LAMBDAH_CONST, LAMBDAH2_CONST,   &
        ZKMENV_CONST, ZKHENV_CONST, PRANDTLSFC_CONST, PRANDTLRAD_CONST, &
        BETA_SURF_CONST, BETA_RAD_CONST, TPFAC_SFC_CONST,   &
        ENTRATE_SFC_CONST, PCEFF_SFC_CONST, KHRADFAC_CONST, &
        KHSFCFAC_CONST, LAMBDA_B_CONST, C_B_CONST, KPBLMIN_CONST
  use LockEntrain, only: ENTRAIN
```

PUBLIC MEMBER FUNCTIONS:

```
  public SetServices
```

DESCRIPTION:

`GEOS_TurbulenceGridComp` computes atmospheric tendencies due to turbulence. Its physics is a combination of the first-order scheme of Louis—for stable PBLs and free atmospheric turbulence—with a modified version of the non-local-K scheme proposed by Lock for unstable and cloud-topped boundary layers. In addition to diffusive tendencies, it adds the effects orographic form drag for features with horizontal scales of 2 to 20 km following Beljaars et al. (2003, ECMWF Tech. Memo. 427).

Grid Considerations

Like all GEOS_Generic-based components, it works on an inherited 3-dimensional ESMF grid. It assumes that the first two (inner) dimensions span the horizontal and the third (outer) dimension is the vertical. In the horizontal, one or both dimensions can be degenerate, effectively supporting single-columns (1-D), and slices (2-D). No horizontal dimension needs to be aligned with a particular coordinate. In the vertical, the only assumption is that columns are indexed from top to bottom.

Methods

`GEOS_TurbulenceGridComp` uses the default Initialize and Finalize methods of GEOS_Generic. It has a 2-stage Run method that can be used in conjunction with two-stage surface calculations to implement semi-implicit time differencing.

Time Behavior

`GEOS_TurbulenceGridComp` assumes both run stages will be invoked every RUN_DT seconds, where RUN_DT is required in the configuration. On this interval both run stages will perform diffusion updates using diffusivities found in the internal state. The diffusivities in the internal state may be refreshed intermitently by specifying MY_STEP and ACCUMINT in the configuration. Accumulated imports used in the intermittent refreshing are valid only on MY_STEP intervals. Currently the origin of these intervals is the beginning of the run. Accumulation of these imports is done for a period ACCUMINT prior to the valid time. Both ACCUMINT and MY_STEP are in seconds.

Working with Bundles and Friendlies

`GEOS_TurbulenceGridComp` works on bundles of quantities to be diffused and with corresponding bundles of their tendencies, surface values, etc. These bundles may contain an arbitrary number of conservative quantities and no requirements or restrictions are placed on what quantities they contain. Quantities required for the calculation, such as pressures, stability, etc are passed separately from the diffused quantities. Little distinction is made of what is in the bundle, except that needed to decide what diffusivity applies to the quantity and in what form its effects are implemented.

Quantities to be diffused can be marked as "Friendly-for-diffusion". In that case, `GEOS_TurbulenceGridComp` directly updates the quantity; otherwise it merely computes its tendency, placing it in the appropriate bundle and treating the quantity itself as read-only.

In working with bundled quantities, corresponding fields must appear in the same order in all bundles. Some of these fields, however, may be "empty" in the sense that the data pointer has not been allocated.

`GEOS_TurbulenceGridComp` works with six bundles; three in the import state and three in the export state. The import bundles are:

TR The quantity being diffused.

TRG The surface (ground) value of the quantity being diffused. (Used only by Run2)

DTG The change of TRG during the time step. (Used only by Run2)

The export bundles are:

TRI The tendency of the quantity being diffused. (Produced by Run1, updated by Run2.)

FSTAR After Run1, the “preliminary” (i.e., at the original surface value) surface flux of the diffused quantity; after Run2, its final value. (Produced by Run1, updated by Run2)

DFSTAR The change of preliminary FSTAR per unit change in the surface value. (Produced by Run1)

All fields in the export bundles are checked for associated pointers before being updated. Fields in the TR bundle can have four attributes:

- FriendlyTo[*Component Name*]: default = false — If true, TR field is updated.
- WeightedTendency: default = true — If true, tendencies (TRI) are pressure-weighted
- DiffuseLike: ('S','Q','M') default = 'S' — Use mixing coefficients for either heat, moisture or momentum.

Only fields in the TR bundle are checked for friendly status. Non-friendly fields in TR and all other bundles are treated with the usual Import/Export rules.

Other imports and exports
In addition to the updates of these bundles, `GEOS_TurbulenceGridComp` produces a number of diagnostic exports, as well as frictional heating contributions. The latter are NOT added by `GEOS_TurbulenceGridComp`, but merely exported to be added elsewhere in the GCM.

Two-Stage Interactions with the Surface
The two-stage scheme for interacting with the surface module is as follows:

- The first run stage takes the surface values of the diffused quantities and the surface exchange coefficients as input. These are, of course, on the grid turbulence is working on.
- It then does the full diffusion calculation assuming the surface values are fixed, i.e., the explicit surface case. In addition, it also computes derivatives of the tendencies wrt surface values. These are to be used in the second stage.
- The second run stage takes the increments of the surface values as inputs and produces the final results, adding the implicit surface contributions.
- It also computes the frictional heating due to both implicit and explicit surface contributions.

GEOS-5 Specific Aspects
In GEOS-5, GEOS_TurbulenceGridComp works on the atmosphere's lat-lon grid, while surface quantities are computed during the first run stage of the each of the tiled surface components. The tiled quantities are properly aggregated to the GEOS-5 lat-lon grid by the first stage of GEOS_SurfaceGridComp, which is called immediately before the first run stage of GEOS_TurbulenceGridComp.

37.1 SetServices – Sets ESMF services for this component

DESCRIPTION:
This version uses the GEOS_GenericSetServices, which sets the Initialize and Finalize services to generic versions. It also allocates our instance of a generic state and puts it in the gridded component (GC). Here we only set the two-stage run method and declare the data services.
REVISION HISTORY:

```
??Jul2006 E.Novak./Todling - Added output defining TLM/ADM trajectory
```

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer, optional                  :: RC  ! return code
```

STATES:

The following is a list of Import, Export and Internal states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
PLE	IM	Pa	xyz	Edge	air pressure
T	IM	K	xyz	Center	air temperature
TH	IM	K	xyz	Center	potential temperature
QV	IM	kg kg^{-1}	xyz	Center	specific humidity
QLLS	IM	kg kg^{-1}	xyz	Center	liquid condensate mixing ratio
QILS	IM	kg kg^{-1}	xyz	Center	frozen condensate mixing ratio
CLLS	IM	1	xyz	Center	cloud fraction
QLCN	IM	kg kg^{-1}	xyz	Center	liquid condensate mixing ratio
QICN	IM	kg kg^{-1}	xyz	Center	frozen condensate mixing ratio
CLCN	IM	1	xyz	Center	cloud fraction
U	IM	m s^{-1}	xyz	Center	eastward wind

Short Name	Type	Units	Dims	Vert Loc	Long name
V	IM	m s^{-1}	xyz	Center	northward wind
CT	IM	kg m^{-2} s^{-1}	xy	None	surface heat exchange coefficient
CQ	IM	kg m^{-2} s^{-1}	xy	None	surface moisture exchange coefficient
CM	IM	kg m^{-2} s^{-1}	xy	None	surface momentum exchange coefficient
BSTAR	IM	m s^{-2}	xy	None	surface bouyancy scale
USTAR	IM	m s^{-1}	xy	None	surface velocity scale
FRLAND	IM	1	xy	None	land fraction
RADLW	IM	K s^{-1}	xyz	Center	air temperature tendency due to longwave
RADLWC	IM	K s^{-1}	xyz	Center	clearsky air temperature tendency lw
PREF	IM	Pa	z	Edge	reference air pressure
VARFLT	IM	m^2	xy	None	variance of filtered topography
TR	IM	X	xyz	Center	diffused quantities
TRG	IM	X	xy	None	surface values of diffused quantity
DTG	IM	X	xy	None	change of surface values of diffused quantity
TRI	EX	X kg m^{-2} s^{-1}	xyz	Center	diffusion tendencies
FSTAR	EX	X kg m^{-2} s^{-1}	xy	None	surface fluxes
DFSTAR	EX	kg m^{-2} s^{-1}	xy	None	change of surface fluxes for unit change of surface value
T	EX	K	xyz	Center	air temperature
U	EX	m s^{-1}	xyz	Center	eastward wind
V	EX	m s^{-1}	xyz	Center	northward wind
QV	EX	kg kg^{-1}	xyz	Center	specific humidity
KM	EX	m^2 s^{-1}	xyz	Edge	total momentum diffusivity
KH	EX	m^2 s^{-1}	xyz	Edge	total scalar diffusivity
RI	EX	1	xyz	Edge	Richardson number from Louis
DU	EX	s^{-1}	xyz	Edge	bulk shear from Louis
KHLS	EX	m^2 s^{-1}	xyz	Edge	scalar diffusivity from Louis
KMLS	EX	m^2 s^{-1}	xyz	Edge	momentum diffusivity from Louis
KHSFC	EX	m^2 s^{-1}	xyz	Edge	surface driven scalar diffusivity from Lock scheme
KHRAD	EX	m^2 s^{-1}	xyz	Edge	radiation driven scalar diffusivity from Lock scheme

Short Name	Type	Units	Dims	Vert Loc	Long name
LWCRT	EX	K s^{-1}	xyz	Center	cloudy LW radiation tendency used by Lock scheme
EKH	EX	m^2 s^{-1}	xyz	Edge	entrainment heat diffusivity from Lock
EKM	EX	m^2 s^{-1}	xyz	Edge	entrainment momentum diffusivity from Lock
ALH	EX	m	xyz	Edge	Blackadar length scale for scalars
INTDIS	EX	K s^{-1} Pa	xyz	Center	p-weighted frictional heating rate from diffusion
TOPDIS	EX	K s^{-1} Pa	xyz	Center	p-weighted frictional heating rate from orographic drag
SRFDIS	EX	K s^{-1} Pa	xy	None	p-weighted frictional heating rate from surface drag
KETRB	EX	W m^{-2}	xy	None	vertically integrated kinetic energy tendency across turbulence
KESRF	EX	W m^{-2}	xy	None	vertically integrated kinetic energy dissipation due to surface friction
KEINT	EX	W m^{-2}	xy	None	vertically integrated kinetic energy dissipation due to diffusion
KETOP	EX	W m^{-2}	xy	None	vertically integrated kinetic energy dissipation due to topographic friction
WESFC	EX	m s^{-1}	xy	None	entrainment velocity from surface plume
WERAD	EX	m s^{-1}	xy	None	entrainment velocity from radiation
WEBRV	EX	m s^{-1}	xy	None	entrainment velocity from buoy rev
DBUOY	EX	m s^{-2}	xy	None	Buoyancy jump across inversion
VSCSFC	EX	m s^{-1}	xy	None	turbulent velocity scale for sfc
VSCRAD	EX	m s^{-1}	xy	None	turbulent velocity scale for cooling
VSCBRV	EX	m s^{-1}	xy	None	turbulent velocity scale for buoy rev
KERAD	EX	m^2 s^{-1}	xy	None	turbulent entrainment diff from cooling
CLDRF	EX	W m^{-2}	xy	None	cloud top radiative forcing
PPBL	EX	Pa	xy	None	pbltop pressure

Short Name	Type	Units	Dims	Vert Loc	Long name
ZSML	EX	m	xy	None	pbltop height for sfc plume LOCK
ZRADML	EX	m	xy	None	depth for rad/brv plume LOCK
ZRADBS	EX	m	xy	None	hght of base for rad/brv plume LOCK
ZCLD	EX	m	xy	None	pbltop cloud depth LOCK
ZCLDTOP	EX	m	xy	None	pbltop cloud top height LOCK
CHIS	EX	1	xy	None	optimal mixture fraction for BRV
SMIXT	EX	J kg^{-1}	xy	None	s of optimal mixture for BRV
DELSINV	EX	K	xy	None	Scaled Del s at Cloud top
DSIEMS	EX	1	xy	None	Siems buoy rev parameter
RADRCODE	EX	1	xy	None	Return codes for Lock top driven plume
AKSODT	EX	1	xyz	Center	matrix diagonal ak for scalars over dt
CKSODT	EX	1	xyz	Center	matrix diagonal ck for scalars over dt
AKQODT	EX	1	xyz	Center	matrix diagonal ak for moisture over dt
CKQODT	EX	1	xyz	Center	matrix diagonal ck for moisture over dt
AKVODT	EX	1	xyz	Center	matrix diagonal ak for winds over dt
CKVODT	EX	1	xyz	Center	matrix diagonal ck for winds over dt
TCZPBL	EX	m	xy	None	transcom planetary boundary layer height
ZPBL2	EX	m	xy	None	planetary boundary layer height threshold 2
ZPBL10p	EX	m	xy	None	planetary boundary layer height threshold 10p
ZPBLHTKE	EX	m	xy	None	planetary boundary layer height horiz tke
ZPBLRI	EX	m	xy	None	planetary boundary layer height rich 0
ZPBLRI2	EX	m	xy	None	planetary boundary layer height rich 02
ZPBLTHV	EX	m	xy	None	planetary boundary layer height thetav
KPBL	EX	1	xy	None	pbltop level
AKS	IN	1	xyz	Center	matrix diagonal ahat for scalars

Short Name	Type	Units	Dims	Vert Loc	Long name
BKS	IN	1	xyz	Center	matrix diagonal bhat for scalars
CKS	IN	1	xyz	Center	matrix diagonal c for scalars
DKS	IN	s^{-1}	xyz	Center	sensitivity of tendency to surface value for scalars
AKQ	IN	1	xyz	Center	matrix diagonal ahat for moisture
BKQ	IN	1	xyz	Center	matrix diagonal bhat for moisture
CKQ	IN	1	xyz	Center	matrix diagonal c for moisture
DKQ	IN	s^{-1}	xyz	Center	sensitivity of tendency to surface value for moisture
AKV	IN	1	xyz	Center	matrix diagonal ahat for winds
BKV	IN	1	xyz	Center	matrix diagonal bhat for winds
CKV	IN	1	xyz	Center	matrix diagonal c for winds
DKV	IN	s^{-1}	xyz	Center	sensitivity of tendency to surface value for winds
EKV	IN	Pa s^{-1}	xyz	Center	momentum mixing factor
FKV	IN	Pa s^{-1}	xyz	Center	topographic roughness factor
ZLE	IN	m	xyz	Edge	geopotential height above surface
SINC	IN	m^2 s^{-3}	xyz	Center	turbulence tendency for dry static energy
ZPBL	IN	m	xy	None	planetary boundary layer height

37.2 RUN1 – First run stage for the MAPL_TurbulenceGridComp component

INTERFACE:

```
subroutine RUN1 ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC
type(ESMF_State),    intent(inout) :: IMPORT
type(ESMF_State),    intent(inout) :: EXPORT
type(ESMF_Clock),    intent(inout) :: CLOCK
integer, optional,   intent(  out) :: RC
```

DESCRIPTION:

The first run stage of `GEOS_TurbulenceGridComp` computes the diffusivities, sets-up the matrix for a backward-implicit computation of the surface fluxes, and solves this system for a fixed surface value of the diffused quantity. Run1 takes as inputs the surface exchange coefficients (i.e., $\rho|U|C_{m,h,q}$) for momentun, heat, and moisture, as well as the pressure, temperature, moisture, and winds for the sounding. These are used only for computing the diffusivities and, as explained above, are not the temperatures, moistures, etc. being diffused.
The computation of turbulence fluxes for fixed surface values is done at every time step in the contained subroutine `DIFFUSE`; but the computation of diffusivities and orographic drag coefficients, as well as the set-up of the vertical difference matrix and its LU decomposition can be done intermittently for economy in the contained subroutine `REFRESH`. The results of this calculation are stored in an internal state. Run1 also computes the sensitivity of the atmospheric tendencies and the surface flux to changes in the surface value.
The diffusivities are computed by calls to `LOUIS_KS` and `ENTRAIN`, which compute the Louis et al. (1983) and Lock (2000) diffusivities. The Louis diffusivities are computed for all conditions, and `ENTRAIN` overrides them where appropriate. Lock can be turned off from the resource file.

37.2.1 REFRESH – Refreshes diffusivities.

INTERFACE:

```
subroutine REFRESH(IM,JM,LM,RC)
```

ARGUMENTS:

```
integer,           intent(IN)       :: IM,JM,LM
integer, optional, intent(OUT)      :: RC
```

DESCRIPTION:
`REFRESH` can be called intermittently to compute new values of the diffusivities. In addition it does all possible calculations that depend only on these. In particular, it sets up the semi-implicit tridiagonal solver in the vertical and does the LU decomposition. It also includes the local effects of orographic drag, so that it to is done implicitly.
Diffusivities are first computed with the Louis scheme (`LOUIS_KS`), and then, where appropriate, they are overridden by the Lock values (`ENTRAIN`). Once diffusivities are computed, `REFRESH` sets-up the tridiagonal matrices for the semi-implicit vertical diffusion calculation and performs their *LU* decomposition.
`REFRESH` requires surface exchange coefficients for heat, moisture, and momentum, The calculations in the interior are also done for momentum, heat, and water diffusion. Heat and water mixing coefficients differ only at the surface, but these affect the entire *LU* decomposition, and so all three decompositions are saved in the internal state.
For a conservatively diffused quantity q, we have

$$\frac{\partial q}{\partial t} = -g\frac{\partial}{\partial p}\left(\rho K_q \frac{\partial q}{\partial z}\right)$$

In finite difference form, using backward time differencing, this becomes

$$
\begin{aligned}
q_l^{n+1} - q_l^n &= -\frac{g}{\delta_l p}^* \delta_l \left[\left(\frac{\Delta t \rho K_q}{\delta_l z} \right)^* (\delta_l q)^{n+1} \right] \\
&= -\alpha_l (\beta_{l+\frac{1}{2}} (q_{l+1} - q_l)^{n+1} - \beta_{l-\frac{1}{2}} (q_l - q_{l-1})^{n+1}) \\
\alpha_l &= \frac{g \Delta t}{(p_{l+\frac{1}{2}} - p_{l-\frac{1}{2}})^*} \\
\beta_{l+\frac{1}{2}} &= \left(\frac{(\rho K_q)^*_{l+\frac{1}{2}}}{(z_{l+1} - z_l)^*} \right)
\end{aligned}
$$

where the subscripts denote levels, superscripts denote times, and the $*$ superscript denotes evaluation at the refresh time. The following tridiagonal set is then solved for q_l^{n+1}:

$$a_l q_{l-1} + b_l q_l + c_l q_{l+1} = q_l$$

where

$$
\begin{aligned}
a_l &= \alpha_l \beta_{l-\frac{1}{2}} \\
c_l &= \alpha_l \beta_{l+\frac{1}{2}} \\
b_l &= 1 - a_l - c_l.
\end{aligned}
$$

At the top boundary, we assume $K_q = 0$, so $\beta_{\frac{1}{2}} = 0$ and $a_1 = 0$. At the surface, $\beta_{L+\frac{1}{2}} = \rho_s |U|_s C_{m,h,q}$, the surface exchange coefficient.

37.2.2 DIFFUSE – Solves for semi-implicit diffusive tendencies assuming fixed surface conditions.

INTERFACE:

```
subroutine DIFFUSE(IM,JM,LM,RC)
```

ARGUMENTS:

```
integer,           intent(IN)       :: IM,JM,LM
integer, optional, intent(OUT)      :: RC
```

DESCRIPTION:

`DIFFUSE` computes semi-implicit tendencies of all fields in the TR bundle. Each field is examined for three attributes: `DiffuseLike`, `FriendlyToTURBULENCE`, and `WeightedTendency`. These determine the behavior of `DIFFUSE` for that field. `DiffuseLike` can be either 'U', 'Q', or 'S'; the default is 'Q'. `FriendlyToTURBULENCE`, and `WeightedTendency` are ESMF logicals. If `FriendlyToTURBULENCE` is true, the field in TR is updated directly; otherwise it is left untouched. In either case, If the corresponding pointer TRI bundle is associated, the tendencies are returned there. If `WeightedTendency` is true, the tendency in TRI, if any, is pressure weighted.

37.3 RUN2 – The second run stage for the TURBULENCE component

INTERFACE:

```
subroutine RUN2 ( GC, IMPORT, EXPORT, CLOCK, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Gridded component
type(ESMF_State),    intent(inout) :: IMPORT ! Import state
type(ESMF_State),    intent(inout) :: EXPORT ! Export state
type(ESMF_Clock),    intent(inout) :: CLOCK  ! The clock
integer, optional,   intent(  out) :: RC     ! Error code:
```

DESCRIPTION:
Second run stage of `GEOS_TurbulenceGridComp` performs the updates due to changes in surface quantities. Its input are the changes in surface quantities during the time step. It can also compute the frictional dissipation terms as exports, but these are not added to the temperatures.

37.3.1 UPDATE – Updates diffusive effects for changes at surface.

INTERFACE:

```
subroutine UPDATE(IM,JM,LM,LATS,RC)
```

ARGUMENTS:

```
integer,           intent(IN)       :: IM,JM,LM
integer, optional, intent(OUT)      :: RC
```

DESCRIPTION:
Some description

37.4 GETKS – Computes atmospheric diffusivities at interior levels

INTERFACE:

```
attributes(global) &
subroutine LOUIS_KS(IRUN,LM,     &
      KH,KM,RI,DU,ZPBL,          &
      ZZ,ZE,PV,UU,VV,            &
      LOUIS, MINSHEAR, MINTHICK, &
      LAMBDAM, LAMBDAM2,         &
```

```
LAMBDAH, LAMBDAH2,         &
ZKMENV, ZKHENV,            &
AKHMMAX,                   &
ALH_DIAG,KMLS_DIAG,KHLS_DIAG)
```

ARGUMENTS:

```
integer, value :: IRUN,LM
integer, intent(IN   ) :: IRUN,LM
real,    intent(IN   ) :: LOUIS         ! Louis scheme parameters (usually 5).
real,    intent(IN   ) :: MINSHEAR      ! Min shear allowed in Ri calculation (s-1).
real,    intent(IN   ) :: MINTHICK      ! Min layer thickness (m).
real,    intent(IN   ) :: AKHMMAX       ! Maximum allowe diffusivity (m+2 s-1).
real,    intent(IN   ) :: LAMBDAM       ! Blackadar(1962) length scale parameter for mom
real,    intent(IN   ) :: LAMBDAM2      ! Second Blackadar parameter for momentum (m).
real,    intent(IN   ) :: LAMBDAH       ! Blackadar(1962) length scale parameter for hea
real,    intent(IN   ) :: LAMBDAH2      ! Second Blackadar parameter for heat (m).
real,    intent(IN   ) :: ZKMENV        ! Transition height for Blackadar param for mome
real,    intent(IN   ) :: ZKHENV        ! Transition height for Blackadar param for heat
real,    intent(IN   ) :: ZZ(IRUN,LM)   ! Height of layer center above the surface (m).
real,    intent(IN   ) :: PV(IRUN,LM)   ! Virtual potential temperature at layer center
real,    intent(IN   ) :: UU(IRUN,LM)   ! Eastward velocity at layer center (m s-1).
real,    intent(IN   ) :: VV(IRUN,LM)   ! Northward velocity at layer center (m s-1).
real,    intent(IN   ) :: ZE(IRUN,LM+1) ! Height of layer base above the surface (m).

! These are 1:LM+1 here but 0:LM in the GC
! Old code only passed in 1:LM-1 from GC which is 2:LM here.
real,    intent(  OUT) :: KM(IRUN,LM+1) ! Momentum diffusivity at base of each layer (m
real,    intent(  OUT) :: KH(IRUN,LM+1) ! Heat diffusivity at base of each layer  (m+2 s
real,    intent(  OUT) :: RI(IRUN,LM+1) ! Richardson number
real,    intent(  OUT) :: DU(IRUN,LM+1) ! Magnitude of wind shear (s-1).
real,    intent(IN   ) :: ZPBL(IRUN)    ! PBL Depth (m)

real,    intent(  OUT) :: ALH_DIAG(IRUN,LM+1)  ! Blackadar Length Scale diagnostic (m)
real,    intent(  OUT) :: KMLS_DIAG(IRUN,LM+1) ! Momentum diffusivity at base of each la
real,    intent(  OUT) :: KHLS_DIAG(IRUN,LM+1) ! Heat diffusivity at base of each layer
```

DESCRIPTION:

Computes Louis et al.(1979) Richardson-number-based diffusivites, as well as an additional "entrainment" diffusivity. The Louis diffusivities for momentum, K_m, and for heat and moisture, K_h, are defined at the interior layer edges. For LM layers, we define diffusivities at the base of the top LM-1 layers. All indexing is from top to bottom of the atmosphere.

The Richardson number, Ri, is defined at the same edges as the diffusivities.

$$\mathrm{Ri}_l = \frac{\frac{g}{\left(\overline{\theta_v}\right)_l}\left(\frac{\delta\theta_v}{\delta z}\right)_l}{\left(\frac{\delta|\mathbf{V}|}{\delta z}\right)_l^2}, \ l = 1, LM - 1$$

where $\theta_v = \theta(1 + \epsilon q)$ is the virtual potential temperature, $\epsilon = \frac{M_a}{M_w} - 1$, M_a and M_w are the molecular weights of dry air and water, and q is the specific humidity. $\delta\theta_v$ is the difference of θ_v in the layers above and below the edge at which Ri_l is defined; $\overline{\theta_v}$ is their average. The diffusivities at the layer edges have the form:

$$K_l^m = (\ell_m^2)_l \left(\frac{\delta|\mathbf{V}|}{\delta z}\right)_l f_m(\mathrm{Ri}_l)$$

and

$$K_l^h = (\ell_h^2)_l \left(\frac{\delta|\mathbf{V}|}{\delta z}\right)_l f_h(\mathrm{Ri}_l),$$

where k is the Von Karman constant, and ℓ is the Blackdar(1962) length scale, also defined at the layer edges.

Different turbulent length scales can be used for heat and momentum. in both cases, we use the traditional formulation:

$$(\ell_{(m,h)})_l = \frac{k z_l}{1 + \frac{k z_l}{\lambda_{(m,h)}}},$$

where, near the surface, the scale is proportional to z_l, the height above the surface of edge level l, and far from the surface it approaches λ. The length scale λ is usually taken to be a constant (order 150 m), assuming the same scale for the outre boundary layer and the free atmosphere. We make it a function of height, reducing its value in the free atmosphere. The momentum length scale written as:

$$\lambda_m = \max(\lambda_1 e^{\left(\frac{z_l}{z_T}\right)^2}, \lambda_2)$$

where $\lambda_2 \leq \lambda_1$ and z_T is the top of the boundary layer. The length scale for heat and other scalers is taken as: $\lambda_h = \sqrt{\frac{3d}{2}}\lambda_m$, following the scheme used at ECMWF.

The two universal functions of the Richardson number, f_m and f_h, are taken from Louis et al (1982). For unstable conditions (Ri$\leq$ 0), they are:

$$f_m = (1 - 2b\psi)$$

and

$$f_h = (1 - 3b\psi),$$

where

$$\psi = \frac{\mathrm{Ri}}{1 + 3bC(z)\sqrt{-\mathrm{Ri}}},$$

and

$$C(z) =$$

For stable condition (Ri$\geq$ 0), they are

$$f_m = \frac{1}{1.0 + \frac{2b\mathrm{Ri}}{\psi}}$$

and

$$f_h = \frac{1}{1.0 + 3b\mathrm{Ri}\psi},$$

where

$$\psi = \sqrt{1 + d\mathrm{Ri}}.$$

As in Louis et al (1982), the parameters appearing in these are taken as $b = c = d = 5$.

Orograpghic drag follows Beljaars (2003):

$$\frac{\partial}{\partial z}\frac{\tau}{\rho} = \frac{C_B}{\lambda_B}|U(z)|U(z)e^{-\tilde{z}^{\frac{3}{2}}}\tilde{z}^{-1.2},$$

where z is the height above the surface in meters, $\tilde{z} = \frac{z}{\lambda_B}$, τ is the orographic stress at z, ρ is the air density, $U(z)$ is the wind velocity, and λ_B is a vertical length scale. Beljaars uses $\lambda_B = 1500$m, for which the non-dimensional parameter $C_B = 2.5101471 \times 10^{-8}$. These are the default values, but both can be modified from the configuration. To avoid underflow. the tendency is set to zero once $\tilde{z}$ exceeds 4 (i.e., 6 km from the surface for default values).

37.5 VTRILU – Does LU decomposition of tridiagonal matrix.

INTERFACE:

```
attributes(global) &
subroutine VTRILU(IRUN,LM,A,B,C)
```

ARGUMENTS:

```
integer, value :: IRUN, LM
integer,                      intent(IN   ) :: IRUN, LM
real,    dimension(IRUN,LM), intent(IN   ) :: C
real,    dimension(IRUN,LM), intent(INOUT) :: A, B
```

DESCRIPTION:

`VTRILU` performs an LU decomposition on a tridiagonal matrix $M = LU$.

$$M = \begin{pmatrix} b_1 & c_1 & & & & & \\ a_2 & b_2 & c_2 & & & & \\ & \cdot & \cdot & \cdot & & & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & a_{K-1} & b_{K-1} & c_{K-1} \\ & & & & & a_K & b_K \end{pmatrix}$$

$$L = \begin{pmatrix} 1 & & & & & \\ \hat{a}_2 & 1 & & & & \\ & \cdot & \cdot & & & \\ & & \cdot & \cdot & & \\ & & & \cdot & \cdot & \\ & & & & \hat{a}_{K-1} & 1 & \\ & & & & & \hat{a}_K & 1 \end{pmatrix} \quad U = \begin{pmatrix} \hat{b}_1 & c_1 & & & & \\ & \hat{b}_2 & c_2 & & & \\ & & \cdot & \cdot & & \\ & & & \cdot & \cdot & \\ & & & & \cdot & \cdot \\ & & & & & \hat{b}_{K-1} & c_{K-1} \\ & & & & & & \hat{b}_K \end{pmatrix}$$

On input, A, B, and C contain, a_k, b_k, and c_k the lower, main, and upper diagonals of the matrix, respectively. On output, B contains $1/\hat{b}_k$, the inverse of the main diagonal of U, and A contains $\hat{a}_k$, the lower diagonal of L. C contains the upper diagonal of the original matrix and of U.
The new diagonals $\hat{a}_k$ and $\hat{b}_k$ are:

$$\begin{aligned} \hat{b}_1 &= b_1, & \\ \hat{a}_k &= a_k/\hat{b}_{k-1}, & k = 2, K, \\ \hat{b}_k &= b_k - c_{k-1}\hat{a}_k, & k = 2, K. \end{aligned}$$

37.6 VTRISOLVESURF – Solves for sensitivity to surface value

INTERFACE:

```
attributes(global) &
subroutine VTRISOLVESURF(IRUN,LM,B,C,Y)
```

ARGUMENTS:

```
integer, value :: IRUN, LM
integer,                      intent(IN   ) :: IRUN, LM
real,    dimension(IRUN,LM), intent(IN   ) :: B, C
real,    dimension(IRUN,LM), intent(  OUT) :: Y
```

DESCRIPTION:
Solves tridiagonal system that has been LU decomposed for the special case where the surface Y (YG) is 1 and the rest of the input Ys are 0. Everything else is as in `VTRISOLVE`. This gives the sensitivity of the solution to a unit change in the surface values.

37.7 VTRISOLVE – Solves for tridiagonal system that has been decomposed by VTRILU

INTERFACE:

```
subroutine VTRISOLVE ( A,B,C,Y,YG )
```

ARGUMENTS:

```
real, dimension(:,:,:),  intent(IN   ) ::  A, B, C
real, dimension(:,:,:),  intent(INOUT) ::  Y
real, dimension(:,:),    intent(IN)    ::  YG
```

DESCRIPTION:
Solves tridiagonal system that has been LU decomposed $LUx = f$. This is done by first solving $Lg = f$ for g, and then solving $Ux = g$ for x. The solutions are:

$$\begin{aligned} g_1 &= f_1, \\ g_k &= f_k - g_{k-1}\hat{a}_k, \qquad\qquad k = 2, K, \end{aligned}$$

and

$$\begin{aligned} x_K &= g_K/\hat{b}_K, \\ x_k &= (g_k - c_k g_{k+1})/\hat{b}_k, \qquad\qquad k = K-1, 1 \end{aligned}$$

On input A contains the $\hat{a}_k$, the lower diagonal of L, B contains the $1/\hat{b}_k$, inverse of the main diagonal of U, C contains the c_k, the upper diagonal of U. The forcing, f_k is
It returns the solution in the r.h.s input vector, Y. A has the multiplier from the decomposition, B the matrix (U), and C the upper diagonal of the original matrix and of U. YG is the LM+1 (Ground) value of Y.

38 Module GEOS_Vegdyn – child to the "Land" gridded component.

DESCRIPTION:
`GEOS_Vegdyn` is a gridded component that performs the necessary interpolation to provide refreshed values of the dynamic vegetation values prescribed by external data/observations.

There are no imports to this routine. Exports from this routine are the instaneous values of the vegetation parameters on tilespace to be used in other components of the land subroutine. All exports and imports are stored on the tile grid inherited from the parent routine.

I. Parameter Class 1: Time AND spatially dependent parameters from a binary data file

Current list: LAI, GRN

The gridded component stores the surrounding observations of each parameter in the internal state. If the run method discovers that the current internal state does not contain the observed values required to interpolate the values at the current time, it performs the required i/o to refresh the values of the internal state. The first iteration of the run method always has to fill the values. No restart is required by this gridded component for these parameters. (A restart *is* now required for Vegetation Class 3

INTERNALS: ITY

EXPORTS: LAI, GRN
USES:

```
use ESMF
use MAPL_Mod
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

38.1 SetServices – Sets ESMF services for this component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
integer, optional                  :: RC  ! return code
```

DESCRIPTION:
This version uses the MAPL_GenericSetServices. This function sets the Initialize and Finalize services, as well as allocating our instance of a generic state and putting it in the gridded component (GC). Here we only need to set the run method and add the state variable specifications (also generic) to our instance of the generic state. This is the way our true state variables get into the ESMF_State INTERNAL, which is in the MAPL_MetaComp.
STATES:

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
ITY	IN	1	tile	None	vegetation type
LAI	EX	1	tile	None	leaf area index
GRN	EX	1	tile	None	greeness fraction
Z2CH	EX	m	tile	None	canopy height
ROOTL	EX	m^2	tile	None	root length density

39 Module GMIchem_GridCompMod - The GMI COMBO Model Grid Component

INTERFACE:

```
MODULE GMIchemGridCompMod
```

USES:

```
USE ESMF
USE MAPL_Mod
USE Chem_Mod                           ! Chemistry Base Class
USE GMI_GridCompMod                              ! ESMF parent component
USE Chem_UtilMod, ONLY : Chem_UtilNegFiller  ! Eliminates negative vmr
USE m_chars, ONLY : uppercase

IMPLICIT NONE
PRIVATE
```

PUBLIC MEMBER FUNCTIONS:

```
PUBLIC SetServices
```

DESCRIPTION:
`GMIchem_GridComp` is a ESMF gridded component for the Global Modeling Initiative combined troposphere/stratospheric chemistry package. REVISION HISTORY:

```
31Jul2006  da Silva  Created the GMI stub.
11Dec2007  Nielsen   Real code for Eros-beta7p17.
25Nov2011  Nielsen   Trying cubed sphere.
10Sep2013  Nielsen   Added run alarm, but allow for updating age-of-air and for
                     returning zero tendencies, etc., when alarm is not ringing.
```

39.1 SetServices — Sets IRF services for GMIchem Grid Component

INTERFACE:

```
SUBROUTINE SetServices ( GC, RC )
```

ARGUMENTS:

```
 type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
 integer, optional                  :: RC  ! return code
```

DESCRIPTION:
Sets Initialize, Run and Finalize services. REVISION HISTORY:

```
31Jul2006  da Silva  First crack.
```

39.2 Initialize_ — Initialize GMIchem

INTERFACE:

```
SUBROUTINE Initialize_ ( gc, impChem, expChem, clock, rc )
```

USES:

INPUT PARAMETERS:

```
type(ESMF_Clock),  intent(inout) :: clock      ! The clock
```

OUTPUT PARAMETERS:

```
type(ESMF_GridComp), intent(inout)  :: gc      ! Grid Component
type(ESMF_State), intent(inout) :: impChem     ! Import State
type(ESMF_State), intent(inout) :: expChem     ! Export State
integer, intent(out) ::  rc                    ! Error return code:
                                               !  0 - all is well
                                               !  1 -
```

DESCRIPTION:
This is a simple ESMF wrapper. REVISION HISTORY:

```
27Feb2005 da Silva  First crack.
```

39.3 Run_ — Runs GMIchem

INTERFACE:

```
SUBROUTINE Run_ ( gc, impChem, expChem, clock, rc )
```

USES:

INPUT PARAMETERS:

```
type(ESMF_Clock),  intent(inout) :: clock      ! The clock
```

OUTPUT PARAMETERS:

```
type(ESMF_GridComp), intent(inout)  :: gc      ! Grid Component
type(ESMF_State), intent(inout) :: impChem     ! Import State
type(ESMF_State), intent(inout) :: expChem     ! Export State
integer, intent(out) ::  rc                    ! Error return code:
                                               !  0 - all is well
                                               !  1 -
```

DESCRIPTION:
This is a simple ESMF wrapper. REVISION HISTORY:

```
27Feb2005 da Silva  First crack.
10Sep2013 Nielsen   Added run alarm, but allow for updating age-of-air and for
                    returning zero tendencies, etc., when alarm is not ringing.
```

39.4 Finalize_ — Finalize GMIchem_GridComp (ESMF)

INTERFACE:

```
SUBROUTINE Finalize_ ( gc, impChem, expChem, clock, rc )
```

USES:

INPUT PARAMETERS:

```
type(ESMF_Clock),  intent(inout) :: clock      ! The clock
```

OUTPUT PARAMETERS:

```
type(ESMF_GridComp), intent(inout)  :: gc      ! Grid Component
type(ESMF_State), intent(inout) :: impChem     ! Import State
type(ESMF_State), intent(inout) :: expChem     ! Export State
integer, intent(out) ::  rc                    ! Error return code:
                                               !  0 - all is well
                                               !  1 -
```

DESCRIPTION:
This is a simple ESMF wrapper. REVISION HISTORY:

```
27Feb2005 da Silva  First crack.
```

40 Module Aero_GridCompMod — Legacy GOCART Grid-Component

INTERFACE:

```
module  AeroGridCompMod
```

USES:

```
use MAPL_Mod, only: MAPL_AM_I_ROOT

use Chem_Mod              ! Chemistry Base Class
use Chem_StateMod         ! Chemistry State
use Chem_MieMod           ! Aerosol LU Tables

Use Chem_UtilMod, only: pmaxmin

use O3_GridCompMod        ! Ozone
use CO_GridCompMod        ! Carbon monoxide
use CO2_GridCompMod       ! Carbon dioxide
use BC_GridCompMod        ! Black Carbon
use DU_GridCompMod        ! Dust
use OC_GridCompMod        ! Organic Carbon
use SS_GridCompMod        ! Sea Salt
use SU_GridCompMod        ! Sulfates
use CFC_GridCompMod       ! CFCs
use Rn_GridCompMod        ! Radon
use CH4_GridCompMod       ! Methane
```

PUBLIC TYPES:

```
PRIVATE
PUBLIC  Aero_GridComp       ! The Legacy GOCART Object
```

PUBLIC MEMBER FUNCTIONS:

```
PUBLIC  Aero_GridCompInitialize
PUBLIC  Aero_GridCompRun
PUBLIC  Aero_GridCompFinalize
```

DESCRIPTION:
This module implements the (pre-ESMF) GOCART Grid Component. This is a composite component which delegates the real work to its sub-components. REVISION HISTORY:

```
16Sep2003 da Silva  First crack.
24Jan2004 da Silva  Added expChem/cdt to interfaces.
24Mar2005 da Silva  Requires RH and saves it under w_c%rh
29Mar2005 da Silva  Initializes AOD LUTs.
18Oct2005 da Silva  Added CO2.
24Jul2006 da Silva  Adapted from Chem_GridComp.
```

40.1 Aero_GridCompInitialize — Initialize Aero_GridComp

INTERFACE:

```
subroutine Aero_GridCompInitialize ( gcThis, w_c, impChem, expChem, &
                                     nymd, nhms, cdt, rc )
```

USES:

INPUT PARAMETERS:

```
type(Chem_Bundle), intent(inout) :: w_c        ! Chemical tracer fields
integer, intent(in) :: nymd, nhms             ! time
real, intent(in) :: cdt                       ! chemistry timestep (secs)
```

OUTPUT PARAMETERS:

```
type(Aero_GridComp), intent(out)  :: gcThis    ! Grid Component
type(ESMF_State), intent(inout) :: impChem     ! Import State
type(ESMF_State), intent(inout) :: expChem     ! Export State
integer, intent(out) ::  rc                    ! Error return code:
                                               !  0 - all is well
                                               !  1 -
```

DESCRIPTION:
Initializes the GOCART Grid Component. It primarily sets the import state for each active constituent package. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

40.2 Aero_GridCompRun — The GOCART Driver

INTERFACE:

```
subroutine Aero_GridCompRun ( gcThis, w_c, impChem, expChem, &
                              nymd, nhms, cdt, rc )
```

USES:

INPUT/OUTPUT PARAMETERS:

```
type(Aero_GridComp), intent(inout) :: gcThis   ! Grid Component
type(Chem_Bundle), intent(inout) :: w_c      ! Chemical tracer fields
```

INPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: impChem  ! Import State
integer, intent(in) :: nymd, nhms          ! time
real, intent(in) :: cdt                    ! chemistry timestep (secs)
```

OUTPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: expChem     ! Export State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This routine implements the so-called GOCART Driver. That is, adds chemical tendencies to each of the constituents, Note: water wapor, the first constituent is not considered a chemical constituents. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
19May2005 da Silva  Phased execution option.
```

40.3 Aero_GridCompFinalize — Finalizes the GOCART Component

INTERFACE:

```
subroutine Aero_GridCompFinalize ( gcThis, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

INPUT/OUTPUT PARAMETERS:

```
type(Aero_GridComp), intent(inout) :: gcThis   ! Grid Component
```

INPUT PARAMETERS:

```
type(Chem_Bundle), intent(in)  :: w_c      ! Chemical tracer fields
integer, intent(in) :: nymd, nhms          ! time
real, intent(in) :: cdt                    ! chemistry timestep (secs)
```

OUTPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: impChem ! Import State
type(ESMF_State), intent(inout) :: expChem   ! Export State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This routine finalizes this Grid Component. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

---

41 Module GOCART_GridCompMod - The GOCART Aerosol Grid Component

INTERFACE:

```
Module GOCART_GridCompMod
```

USES:

```
use ESMF
use MAPL_Mod
use MAPL_GenericMod

use Chem_Mod              ! Chemistry Base Class
use Chem_UtilMod, only: Chem_UtilNegFiller
use Aero_GridCompMod      ! Parent Aerosol component with
                          !   IRF methods but no SetServices()
USE m_chars, ONLY: uppercase
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

DESCRIPTION:
GOCART is a gridded component from the GOCART model and includes dust, sea salt, sulfates, organic and black carbon. In addition, we also include closely related components for CO and CO2 with relatively simple parameterization of the chemical processes, but sharing consistent emissions with the aerosols.
This code derives from the pre-ESMF Chem component from GEOS-4. This GEOS-4 Chem "component" used ESMF like constructs (Chem component class, import/export states, etc) but no ESMF specific data types because of an odd incompatibility with the fvGCM code (the so-called oldworld library. Unlike GEOS-4, the Stratospheric Chemistry component is treated separately here. REVISION HISTORY:

```
25feb2005  da Silva  First crack.
19jul2006  da Silva  First separate GOCART component.
```

41.1 SetServices — Sets IRF services for GOCART Grid Component

INTERFACE:

```
subroutine SetServices ( GC, RC )
```

ARGUMENTS:

```
 type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
 integer, optional                  :: RC  ! return code
```

DESCRIPTION:
Sets Initialize, Run and Finalize services. REVISION HISTORY:

```
25feb2005  da Silva  First crack.
```

41.2 Initialize_ — Initialize Aero_GridComp (ESMF)

INTERFACE:

```
subroutine Initialize_ ( gc, impChem, expChem, clock, rc )
```

USES:

INPUT PARAMETERS:

```
type(ESMF_Clock),  intent(inout) :: clock     ! The clock
```

OUTPUT PARAMETERS:

```
type(ESMF_GridComp), intent(inout)  :: gc      ! Grid Component
type(ESMF_State), intent(inout) :: impChem     ! Import State
type(ESMF_State), intent(inout) :: expChem     ! Export State
integer, intent(out) ::  rc                    ! Error return code:
                                               !  0 - all is well
                                               !  1 -
```

DESCRIPTION:
This is a simple ESMF wrapper. REVISION HISTORY:

```
27Feb2005 da Silva  First crack.
```

41.3 Run_ — Runs Aero_GridComp (ESMF)

INTERFACE:

```
subroutine Run_ ( gc, impChem, expChem, clock, rc )
```

USES:

INPUT PARAMETERS:

```
type(ESMF_Clock),  intent(inout) :: clock     ! The clock
```

OUTPUT PARAMETERS:

```
type(ESMF_GridComp), intent(inout)  :: gc      ! Grid Component
type(ESMF_State), intent(inout) :: impChem     ! Import State
type(ESMF_State), intent(inout) :: expChem     ! Export State
integer, intent(out) ::  rc                    ! Error return code:
                                               !  0 - all is well
                                               !  1 -
```

DESCRIPTION:
This is a simple ESMF wrapper. REVISION HISTORY:

```
27Feb2005 da Silva  First crack.
```

41.4 Finalize_ — Finalize Aero_GridComp (ESMF)

INTERFACE:

```
subroutine Finalize_ ( gc, impChem, expChem, clock, rc )
```

USES:

INPUT PARAMETERS:

```
type(ESMF_Clock),  intent(inout) :: clock      ! The clock
```

OUTPUT PARAMETERS:

```
type(ESMF_GridComp), intent(inout)  :: gc      ! Grid Component
type(ESMF_State), intent(inout) :: impChem     ! Import State
type(ESMF_State), intent(inout) :: expChem     ! Export State
integer, intent(out) ::  rc                    ! Error return code:
                                               !  0 - all is well
                                               !  1 -
```

DESCRIPTION:
This is a simple ESMF wrapper. REVISION HISTORY:

```
27Feb2005 da Silva  First crack.
```

42 Module MAMchem_GridCompMod - Implements MAM Chemistry

INTERFACE:

```
MODULE MAMchemGridCompMod
```

USES:

```
USE ESMF
USE MAPL_Mod

USE MAPL_SimpleBundleMod

USE MAM3_DataMod
USE MAM7_DataMod

USE MAM_BaseMod,        only: MAM3_MODEL, MAM7_MODEL, MAM_MetaSpec, MAM_MetaInit
```

```
USE MAM_SizeMod,        only: MAM_DrySize, MAM_WetSize

USE MAM_SeasaltMod,     only: MAM_SS_Emission, MAM_SS_Diagnostics
USE MAM_DustMod,        only: MAM_DU_Emission, MAM_DU_Diagnostics

USE MAM_NucleationMod,  only: MAM_Nucleation
USE MAM_CoagulationMod, only: MAM_CoagulationBimodal
USE MAM_DryRemovalMod,  only: MAM_DryRemoval


USE CAM_BaseMod,     only: CAM_Initialize, CAM_CalculateSize

IMPLICIT NONE
PRIVATE
```

PUBLIC MEMBER FUNCTIONS:

```
PUBLIC SetServices
```

DESCRIPTION:
`MAMchem_GridComp` is an ESMF gridded component implementing the MAM aerosol microphysical processes.
Developed for GEOS-5 release Fortuna 2.0 and later. REVISION HISTORY:

```
06Dec2009  da Silva     Created the MATRIX skeleton.
15Aug2011  A. Darmenov  Initial version of MAM
```

42.1 SetServices — Sets IRF services for the MAMchem Grid Component

INTERFACE:

```
SUBROUTINE SetServices ( GC, RC )
```

ARGUMENTS:

```
 type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
 integer, optional                  :: RC  ! return code
```

DESCRIPTION:
Sets Initialize, Run and Finalize services. REVISION HISTORY:

```
1Dec2009  da Silva  First crack.
```

STATES:

The following is a list of `Import`, `Export` and `Internal` states (second column specifies the type):

Short Name	Type	Units	Dims	Vert Loc	Long name
AERO	EX	kg kg^{-1}	xyz	Center	aerosol mass mixing ratios
AERO_DP	EX	kg m^{-2} s^{-1}	xy		aerosol deposition

42.2 Initialize_ — Initialize MAMchem

INTERFACE:

```
SUBROUTINE Initialize_ ( GC, IMPORT, EXPORT, CLOCK, rc )
```

USES:

INPUT PARAMETERS:

```
type(ESMF_Clock),  intent(inout)   :: CLOCK  ! The clock
```

OUTPUT PARAMETERS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Grid Component
type(ESMF_State), intent(inout)    :: IMPORT ! Import State
type(ESMF_State), intent(inout)    :: EXPORT ! Export State
integer, intent(out)               :: rc     ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This is a simple ESMF wrapper. REVISION HISTORY:

```
01Dec2009 da Silva  First crack.
```

42.3 Run_ — Runs MAMchem

INTERFACE:

```
SUBROUTINE Run_ ( GC, IMPORT, EXPORT, CLOCK, rc )
```

USES:

INPUT PARAMETERS:

```
type(ESMF_Clock),  intent(inout)   :: CLOCK  ! The clock
```

OUTPUT PARAMETERS:

```
type(ESMF_GridComp), intent(inout) :: GC     ! Grid Component
type(ESMF_State), intent(inout)    :: IMPORT ! Import State
type(ESMF_State), intent(inout)    :: EXPORT ! Export State
integer, intent(out)               :: rc     ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This is a simple ESMF wrapper. REVISION HISTORY:

```
27Feb2005 da Silva  First crack.
```

42.4 Finalize_ — Finalize MAMchem

INTERFACE:

```
SUBROUTINE Finalize_ ( GC, IMPORT, EXPORT, CLOCK, rc )
```

USES:

INPUT PARAMETERS:

```
type(ESMF_Clock),  intent(inout)   :: CLOCK  ! The clock
```

OUTPUT PARAMETERS:

```
type(ESMF_GridComp), intent(inout) :: gc     ! Grid Component
type(ESMF_State), intent(inout)    :: IMPORT ! Import State
type(ESMF_State), intent(inout)    :: EXPORT ! Export State
integer, intent(out)               :: rc     ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This is a simple ESMF wrapper. REVISION HISTORY:

```
01Dec2009  da Silva  First crack.
```

43 Module O3_GridCompMod

Grid Component class for parameterized Chemistry for ozone: INTERFACE:

```
MODULE  O3GridCompMod
```

USES:

```
USE ESMF
USE MAPL_Mod

USE Chem_Mod       ! Chemistry Base Class
USE Chem_StateMod      ! Chemistry State
Use Chem_UtilMod, ONLY: pmaxmin            ! Utilities
Use Chem_UtilMod, ONLY: Chem_UtilMPread
USE Chem_UtilMod, ONLY: Chem_UtilTroppFixer ! Fixes bad tropopause pressure values
USE m_inpak90      ! Resource file management

USE ESMF_CFIOFileMOD
USE MAPL_CFIOMOD

IMPLICIT NONE
```

PUBLIC TYPES:

```
PRIVATE

PUBLIC  O3_GridComp       ! The O3 object
```

PUBLIC MEMBER FUNCTIONS:

```
PUBLIC  O3_GridCompInitialize
PUBLIC  O3_GridCompRun
PUBLIC  O3_GridCompFinalize
```

DESCRIPTION:
This module implements a parameterized chemistry for ozone that includes dry deposition based on the GMIchem handling. REVISION HISTORY:

```
     2000 Nielsen   Initial coding
  4Mar2005 Nielsen   Implementation of parameterized ozone chemistry
31Jan2011 Nielsen   Add dry deposition and NetCDF reads from PCHEM
```

43.1 O3_GridCompInitialize — Initialize O3_GridComp

INTERFACE:

```
SUBROUTINE O3_GridCompInitialize ( gcO3, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT none
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), INTENT(IN) :: w_c     ! Chemical tracer fields, delp, +
INTEGER, INTENT(IN) :: nymd, nhms     ! time
REAL,    INTENT(IN) :: cdt     ! chemistry time step (secs)
```

OUTPUT PARAMETERS:

```
TYPE(O3_GridComp), INTENT(INOUT) :: gcO3 ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem  ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem  ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
```

DESCRIPTION:
Initializes the O3 Grid Component. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
 4Mar2005 Nielsen   Implementation of parameterized ozone chemistry
31Jan2011 Nielsen   Add dry deposition and NetCDF reads from PCHEM
```

43.2 setUpPandL

INTERFACE:

```
SUBROUTINE setUpPandL(rc)
```

USES:

```
IMPLICIT NONE

!INPUT PARAMETERS
```

OUTPUT PARAMETERS:

```
INTEGER, OPTIONAL, INTENT(OUT) ::  rc  ! Error return code:
```

DESCRIPTION:
Read PCHEM's NetCDF file and distribute. Code borrowed from GEOS_PChemGridComp.F90 with minor modifications. For use with one-year datasets ONLY! REVISION HISTORY:

43.3 O3_GridCompRun — The O3 run method

INTERFACE:

```
SUBROUTINE O3_GridCompRun ( gcO3, w_c, impChem, expChem, &
                            nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT none
```

INPUT/OUTPUT PARAMETERS:

```
TYPE(O3_GridComp), INTENT(INOUT) :: gcO3   ! Grid Component
TYPE(Chem_Bundle), INTENT(INOUT) :: w_c    ! Chemical tracer fields
```

INPUT PARAMETERS:

```
TYPE(ESMF_State), INTENT(INOUT) :: impChem ! Import State
INTEGER, INTENT(IN) :: nymd, nhms       ! time
REAL,    INTENT(IN) :: cdt       ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(ESMF_State), INTENT(INOUT) :: expChem   ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This routine implements a parameterized chemistry for ozone. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
 4Mar2005 Nielsen   Implementation of parameterized ozone chemistry
31Jan2012 Nielsen   Revisions for running dry deposition plagarized
                    from GMIChem
```

43.4 doProdLoss

Run the parameterized chemistry for Ox. Ozone is derived from Ox. INTERFACE:

```
SUBROUTINE doProdLoss(rc)
```

USES:

```
IMPLICIT NONE

 INTEGER, OPTIONAL, INTENT(OUT) :: rc
```

DESCRIPTION:
This module implements a parameterized chemistry for ozone. The NetCDF file that contains the production rates and loss frequencies has coefficients for seven species, OX, N2O, CFC-11, CFC-12, CH4, HCFC-22, and H2O.

Advection produces the "intermediate" constituent distribution before this routine is called.

USAGE NOTES:

The resulting O3 mole fraction is the product of the Ox mole fraction multiplied by the O3-to-Ox ratio, ro3ox. At pressures greater than approximately 1 hPa, ro3ox = 1 everywhere. At pressures less than approximately 0.1 hPa, Ox is mostly O3 at night and ro3ox = 1. During the day, ro3ox in this region depends to first order on pressure.

Code is plagarized from GEOS_PchemGridComp.F90 REVISION HISTORY:

```
31Jan2011 Nielsen
```

43.5 DeposVelo

INTERFACE:

```
SUBROUTINE DeposVelo(npts, ij, radiat, tempk, suncos, f0, hstar, xmw,  &
                     ustar, cz1, obk, cfrac, lsnow, rhoa, dvel, rc)

IMPLICIT NONE

----------------------
Argument declarations.
----------------------
```

ARGUMENTS:

```
INTEGER, INTENT(IN) :: npts, ij
REAL, INTENT(IN)    :: radiat(npts)
REAL, INTENT(IN)    :: tempk (npts)
REAL, INTENT(IN)    :: suncos(npts)
REAL, INTENT(IN)    :: f0
REAL, INTENT(IN)    :: hstar
REAL, INTENT(IN)    :: xmw
REAL, INTENT(IN)    :: ustar (npts) ! Cannot be identically zero
REAL, INTENT(IN)    :: cz1   (npts)
REAL, INTENT(IN)    :: obk   (npts)
REAL, INTENT(IN)    :: cfrac (npts)
INTEGER, INTENT(IN) :: lsnow (npts)
REAL, INTENT(IN)    :: rhoa  (npts)
REAL, INTENT(OUT)   :: dvel  (npts)
INTEGER, INTENT(OUT):: rc
```

DESCRIPTION:
This routine computes the dry deposition velocities using a resistance-in-series model. Routine reads data which:

- converts land type id to deposition surface type id
- gives roughness heights for each land type id
- identifies water land type id's, for stability and z0 calculations
- reads surface resistance data for each deposition surface type id

Changes from version 3.1 to version 3.2:

- In unstable atmospheres with —zlmo— ¡ zo, as can happen occasionally under very low wind conditions with tall canopies, application of Monin-Obukhov similarity yields negative values for ra. This was a problem in version 3.1. In fact, Monin-Obukhov similarity does not apply under such conditions, so we now set ra to zero and let the boundary resistance rb define the overall aerodynamic resistance. Since rb varies inversely with u*, it will impose a large aerodynamic resistance under very low wind conditions.
- The range of applicability of stability correction functions to Monin-Obukhov similarity has been extended to -2.5 ¡ z/zmo ¡ 1.5, based on Figure 2 of Businger et al. [1971]. The range used to be -1 ¡ z/zmo ¡ 1 in version 3.1.

Literature cited:

1. Baldocchi, D.D., B.B. Hicks, and P. Camara, A canopy stomatal resistance model for gaseous deposition to vegetated surfaces, Atmos. Environ. 21, 91-101, 1987.

2. Brutsaert, W., Evaporation into the Atmosphere, Reidel, 1982. Businger, J.A., et al., Flux-profile relationships in the atmospheric surface layer, J. Atmos. Sci., 28, 181-189, 1971.

3. Dwight, H.B., Tables of integrals and other mathematical data, MacMillan, 1957.

4. Guenther, A., and 15 others, A global model of natural volatile organic compound emissions, J. Geophys. Res., 100, 8873-8892, 1995.

5. Hicks, B.B., and P.S. Liss, Transfer of SO2 and other reactive gases across the air-sea interface, Tellus, 28, 348-354, 1976.

6. Jacob, D.J., and S.C. Wofsy, Budgets of reactive nitrogen, hydrocarbons, and ozone over the Amazon forest during the wet season, J. Geophys. Res., 95, 16737-16754, 1990.

7. Jacob, D.J., and 9 others, Deposition of ozone to tundra, J. Geophys. Res., 97, 16473-16479, 1992.

8. Levine, I.N., Physical Chemistry, 3rd ed., McGraw-Hill, New York, 1988.

9. Munger, J.W., and 8 others, Atmospheric deposition of reactive nitrogen oxides and ozone in a temperate deciduous forest and a sub-arctic woodland, J. Geophys. Res., in press, 1996.

10. Walcek, C.J., R.A. Brost, J.S. Chang, and M.L. Wesely, SO2, sulfate, and HNO3 deposition velocities computed using regional landuse and meteorological data, Atmos. Environ., 20, 949-964, 1986.

11. Wang, Y.H., paper in preparation, 1996.

12. Wesely, M.L, Improved parameterizations for surface resistance to gaseous dry deposition in regional-scale numerical models, Environmental Protection Agency Report EPA/600/3-88/025, Research Triangle Park (NC), 1988.

13. Wesely, M.L., same title, Atmos. Environ., 23, 1293-1304, 1989.

43.6 O3_GridCompFinalize — The Chem Driver

INTERFACE:

```
SUBROUTINE O3_GridCompFinalize ( gcO3, w_c, impChem, expChem, &
                                 nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT none
```

INPUT/OUTPUT PARAMETERS:

```
TYPE(O3_GridComp), INTENT(INOUT) :: gcO3   ! Grid Component
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), INTENT(IN)  :: w_c      ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms       ! time
REAL,    INTENT(IN) :: cdt       ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(ESMF_State), INTENT(INOUT) :: impChem ! Import State
TYPE(ESMF_State), INTENT(INOUT) :: expChem ! Import State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This routine finalizes this Grid Component. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
 4Mar2005 Nielsen   Implementation of parameterized ozone chemistry
31Jan2012 Nielsen   Add dry deposition and NetCDF reads from PCHEM
```

44 Module OC_GridCompMod — OC Grid Component Class

INTERFACE:

```
module  OCGridCompMod
```

USES:

```
USE ESMF
USE MAPL_Mod

use Chem_Mod              ! Chemistry Base Class
use Chem_StateMod         ! Chemistry State
use Chem_ConstMod, only: grav, von_karman, cpd, &
                        undefval  = > undef         ! Constants !
use Chem_UtilMod          ! I/O
use Chem_MieMod           ! Aerosol LU Tables, calculator
use m_inpak90             ! Resource file management
use m_die, only: die
```

```
USE m_chars, ONLY: lowercase
use Chem_SettlingMod      ! Settling
use DryDepositionMod
use WetRemovalMod
use ConvectionMod         ! Offline convective mixing/scavenging
```

PUBLIC TYPES:

```
PRIVATE
PUBLIC  OC_GridComp       ! The OC object
PUBLIC  OC_GridComp1      ! Single instance OC object
```

PUBLIC MEMBER FUNCTIONS:

```
PUBLIC  OC_GridCompInitialize
PUBLIC  OC_GridCompRun
PUBLIC  OC_GridCompFinalize
```

DESCRIPTION:
This module implements the (pre-ESMF) OC Grid Component. REVISION HISTORY:

```
16Sep2003 da Silva  First crack.
```

44.1 OC_GridCompInitialize — Initialize OC_GridComp

INTERFACE:

```
subroutine OC_GridCompInitialize ( gcOC, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

INPUT PARAMETERS:

```
type(Chem_Bundle), intent(inout) :: w_c        ! Chemical tracer fields
integer, intent(in) :: nymd, nhms             ! time
real, intent(in) :: cdt                       ! chemistry timestep (secs)
```

OUTPUT PARAMETERS:

```
type(OC_GridComp), intent(inout) :: gcOC   ! Grid Component
type(ESMF_State), intent(inout)  :: impChem  ! Import State
type(ESMF_State), intent(inout)  :: expChem  ! Export State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Initializes the OC Grid Component. It primarily sets the import state for each active constituent package. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

44.2 OC_GridCompRun — Run OC_GridComp

INTERFACE:

```
subroutine OC_GridCompRun ( gcOC, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), intent(in) :: w_c        ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms           ! time
REAL,    INTENT(IN) :: cdt                  ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(OC_GridComp), INTENT(INOUT) :: gcOC     ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem  ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem  ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Runs the CO Grid Component. Multiple instance version. REVISION HISTORY:

```
27Feb2008  da Silva  Introduced multiple instances
```

44.3 OC_GridCompFinalize — Initialize OC_GridComp

INTERFACE:

```
subroutine OC_GridCompFinalize ( gcOC, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), intent(in) :: w_c        ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms           ! time
REAL,    INTENT(IN) :: cdt                  ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(OC_GridComp), INTENT(INOUT) :: gcOC     ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem  ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem  ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Finalizes the OC Grid Component. Multiple instance version. REVISION HISTORY:

```
27Feb2008  da Silva  Introduced multiple instances
```

44.4 OC_GridCompInitialize — Initialize OC_GridComp

INTERFACE:

```
subroutine OC_GridCompInitialize1_ ( gcOC, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

INPUT PARAMETERS:

```
type(Chem_Bundle), intent(inout) :: w_c     ! Chemical tracer fields
integer, intent(in) :: nymd, nhms           ! time
real, intent(in) :: cdt                     ! chemistry timestep (secs)
```

OUTPUT PARAMETERS:

```
type(OC_GridComp1), intent(inout) :: gcOC    ! Grid Component
type(ESMF_State), intent(inout)  :: impChem  ! Import State
type(ESMF_State), intent(inout)  :: expChem  ! Export State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Initializes the OC Grid Component. It primarily sets the import state for each active constituent package. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

44.5 OC_GridCompRun — The Chem Driver

INTERFACE:

```
subroutine OC_GridCompRun1_ ( gcOC, w_c, impChem, expChem, &
                              nymd, nhms, cdt, rc )
```

USES:

INPUT/OUTPUT PARAMETERS:

```
type(OC_GridComp1), intent(inout) :: gcOC   ! Grid Component
type(Chem_Bundle), intent(inout) :: w_c      ! Chemical tracer fields
```

INPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: impChem    ! Import State
integer, intent(in) :: nymd, nhms           ! time
real, intent(in) :: cdt                     ! chemistry timestep (secs)
```

OUTPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: expChem   ! Export State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This routine implements the so-called OC Driver. That is, adds chemical tendencies to each of the constituents, Note: water wapor, the first constituent is not considered a chemical constituents. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

44.6 OC_Emission - Adds Organic Carbon emission for one timestep

We have emissions from 5 sources, which are distributed differently in the vertical 1) biomass burning - uniformly mixed in PBL 2) biofuel sources - emitted into lowest 100 m 3) anthropogenic l1 - emitted into lowest 100 m 4) anthropogenic l2 - emitted into 100 - 500 m levels 5) terpene - emitted to surface (hydrophilic only) INTERFACE:

```
subroutine OC_Emission ( i1, i2, j1, j2, km, nbins, cdt, gcOC, w_c, &
                         pblh, tmpu, rhoa, OC_emis, &
                         OC_emisAN, OC_emisBB, OC_emisBF, OC_emisBG, rc )
```

USES:

INPUT PARAMETERS:

```
integer, intent(in) :: i1, i2, j1, j2, km, nbins
real, intent(in)    :: cdt
type(OC_GridComp1), intent(in)    :: gcOC       ! OC Grid Component
real, pointer, dimension(:,:)    :: pblh
real, pointer, dimension(:,:,:)  :: tmpu
real, pointer, dimension(:,:,:)  :: rhoa
```

OUTPUT PARAMETERS:

```
type(Chem_Bundle), intent(inout) :: w_c         ! Chemical tracer fields
type(Chem_Array), intent(inout)  :: OC_emis(nbins) ! OC emissions, kg/m2/s
type(Chem_Array), intent(inout)  :: OC_emisAN      ! OC emissions, kg/m2/s
type(Chem_Array), intent(inout)  :: OC_emisBB      ! OC emissions, kg/m2/s
type(Chem_Array), intent(inout)  :: OC_emisBF      ! OC emissions, kg/m2/s
type(Chem_Array), intent(inout)  :: OC_emisBG      ! OC emissions, kg/m2/s
```

```
integer, intent(out)           :: rc          ! Error return code:
                                              !  0 - all is well
                                              !  1 -
character(len = *), parameter :: myname  =  'OC_Emission'
```

DESCRIPTION:
Updates the OC concentration with emissions every timestep REVISION HISTORY:

```
06Nov2003, Colarco
Based on Ginoux
```

44.7 OC_Compute_Diags - Calculate dust 2D diagnostics

INTERFACE:

```
subroutine OC_Compute_Diags ( i1, i2, j1, j2, km, nbins, gcOC, w_c, tmpu, rhoa, u, v, &
                              sfcmass, colmass, mass, exttau, scatau, &
                              conc, extcoef, scacoef, angstrom, fluxu, fluxv, rc )
```

USES:

INPUT PARAMETERS:

```
integer, intent(in) :: i1, i2, j1, j2, km, nbins
type(OC_GridComp1), intent(inout):: gcOC     ! OC Grid Component
type(Chem_Bundle), intent(in)   :: w_c      ! Chem Bundle
real, pointer, dimension(:,:,:) :: tmpu     ! temperature [K]
real, pointer, dimension(:,:,:) :: rhoa     ! air density [kg m-3]
real, pointer, dimension(:,:,:) :: u        ! east-west wind [m s-1]
real, pointer, dimension(:,:,:) :: v        ! north-south wind [m s-1]
```

OUTPUT PARAMETERS:

```
type(Chem_Array), intent(inout)  :: sfcmass  ! sfc mass concentration kg/m3
type(Chem_Array), intent(inout)  :: colmass  ! col mass density kg/m2
type(Chem_Array), intent(inout)  :: mass     ! 3d mass mixing ratio kg/kg
type(Chem_Array), intent(inout)  :: exttau   ! ext. AOT at 550 nm
type(Chem_Array), intent(inout)  :: scatau   ! sct. AOT at 550 nm
type(Chem_Array), intent(inout)  :: conc     ! 3d mass concentration, kg/m3
type(Chem_Array), intent(inout)  :: extcoef  ! 3d ext. coefficient, 1/m
type(Chem_Array), intent(inout)  :: scacoef  ! 3d scat.coefficient, 1/m
type(Chem_Array), intent(inout)  :: angstrom ! 470-870 nm Angstrom parameter
```

```
type(Chem_Array), intent(inout)  :: fluxu    ! Column mass flux in x direction
type(Chem_Array), intent(inout)  :: fluxv    ! Column mass flux in y direction
integer, intent(out)             :: rc       ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Calculates some simple 2d diagnostics from the OC fields Surface concentration (dry) Column mass load (dry) Extinction aot 550 (wet) Scattering aot 550 (wet) For the moment, this is hardwired. REVISION HISTORY:

```
16APR2004, Colarco
```

44.8 OC_GridCompFinalize — The Chem Driver

INTERFACE:

```
subroutine OC_GridCompFinalize1_ ( gcOC, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

INPUT/OUTPUT PARAMETERS:

```
type(OC_GridComp1), intent(inout) :: gcOC   ! Grid Component
```

INPUT PARAMETERS:

```
type(Chem_Bundle), intent(in)  :: w_c      ! Chemical tracer fields
integer, intent(in) :: nymd, nhms          ! time
real,    intent(in) :: cdt                 ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: impChem   ! Import State
type(ESMF_State), intent(inout) :: expChem   ! Import State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This routine finalizes this Grid Component. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

44.9 OC_SingleInstance_ — Runs single instance of method

INTERFACE:

```
subroutine OC_SingleInstance_ ( Method_, instance, &
                                gcOC, w_c, impChem, expChem, &
                                nymd, nhms, cdt, rc )
```

USES:

```
Use OC_GridCompMod
Use ESMF
Use MAPL_Mod
Use Chem_Mod

IMPLICIT NONE
```

INPUT PARAMETERS:

```
Input "function pointer"
-----------------------
interface
  subroutine Method_ (gc, w, imp, exp, ymd, hms, dt, rcode )
    Use OC_GridCompMod
    Use ESMF
    Use MAPL_Mod
    Use Chem_Mod
    type(OC_GridComp1),  intent(inout)  :: gc
    type(Chem_Bundle),   intent(in)     :: w
    type(ESMF_State),    intent(inout)  :: imp
    type(ESMF_State),    intent(inout)  :: exp
    integer,             intent(in)     :: ymd, hms
    real,                intent(in)     :: dt
    integer,             intent(out)    :: rcode
  end subroutine Method_
end interface

integer, intent(in)           :: instance   ! instance number

TYPE(Chem_Bundle), intent(inout) :: w_c     ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms           ! time
REAL,    INTENT(IN) :: cdt                  ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(OC_GridComp1), INTENT(INOUT) :: gcOC    ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem  ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem  ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Finalizes the CO Grid Component. Multiple instance version. REVISION HISTORY:

```
27Feb2008  da Silva  Introduced multiple instances
```

45 Module Rn_GridCompMod — Rn Grid Component Class

INTERFACE:

```
MODULE  RnGridCompMod
```

USES:

```
USE ESMF
USE MAPL_Mod

USE Chem_Mod       ! Chemistry Base Class
USE Chem_StateMod      ! Chemistry State
USE Chem_ConstMod, only: grav
USE Chem_UtilMod      ! I/O

USE m_inpak90      ! Resource file management
USE m_die, ONLY: die
USE m_chars, ONLY: lowercase

IMPLICIT NONE
```

PUBLIC TYPES:

```
PRIVATE
PUBLIC  Rn_GridComp       ! Multiple instance Radon object
PUBLIC  Rn_GridComp1      ! Single instance Radon object
```

PUBLIC MEMBER FUNCTIONS:

```
PUBLIC  Rn_GridCompInitialize
PUBLIC  Rn_GridCompRun
PUBLIC  Rn_GridCompFinalize
```

DESCRIPTION:
This module implements the Rn Grid Component. REVISION HISTORY:

```
16Sep2003 da Silva  First crack.
01Aug2006 da Silva  Extensions for GEOS-5.
10Mar2008 da Silva  Multiple instances for ARCTAS.
12Apr2008 Nielsen   Configured for radon.
```

45.1 Rn_GridCompInitialize — Initialize Rn_GridComp

INTERFACE:

```
subroutine Rn_GridCompInitialize ( gcRn, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), intent(in) :: w_c        ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms        ! time
REAL,    INTENT(IN) :: cdt        ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(Rn_GridComp), INTENT(INOUT) :: gcRn   ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem  ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem  ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Initializes the Rn Grid Component. Multiple instance version. REVISION HISTORY:

```
27Feb2008  da Silva  Introduced multiple instances
12Apr2008  Nielsen   Configured for radon
```

45.2 Rn_GridCompRun — Run Rn_GridComp

INTERFACE:

```
SUBROUTINE Rn_GridCompRun ( gcRn, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), intent(in) :: w_c        ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms        ! time
REAL,    INTENT(IN) :: cdt        ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(Rn_GridComp), INTENT(INOUT) :: gcRn   ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem  ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem  ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Runs the Rn Grid Component. Multiple instance version. REVISION HISTORY:

```
27Feb2008  da Silva  Introduced multiple instances
12Apr2008  Nielsen   Configured for radon
```

45.3 Rn_GridCompFinalize — Initialize Rn_GridComp

INTERFACE:

```
SUBROUTINE Rn_GridCompFinalize ( gcRn, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), intent(in) :: w_c        ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms        ! time
REAL,    INTENT(IN) :: cdt        ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(Rn_GridComp), INTENT(INOUT) :: gcRn   ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem  ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem  ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Finalizes the Rn Grid Component. Multiple instance version. REVISION HISTORY:

```
27Feb2008  da Silva  Introduced multiple instances
12Apr2008  Nielsen   Configured for radon
```

45.4 Rn_GridCompInitialize — Initialize Rn_GridComp

INTERFACE:

```
subroutine Rn_GridCompInitialize1_ ( gcRn, w_c, impChem, expChem, &
                                     nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), intent(in) :: w_c        ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms        ! time
REAL,    INTENT(IN) :: cdt        ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(Rn_GridComp1), INTENT(INOUT) :: gcRn   ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem  ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem  ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Initializes the Rn Grid Component. It primarily sets the import state for each active constituent package. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
31May2005  Nielsen  Mods for 7 CO bins, 5 region masks
04Nov2005     Bian  CO tagged to 4 regions
                    (global, North America, South America, and Africa)
                    for CR-AVE
12Apr2008  Nielsen  Configured for radon
```

45.5 Rn_GridCompRun

INTERFACE:

```
SUBROUTINE Rn_GridCompRun1_ ( gcRn, w_c, impChem, expChem, &
                              nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT/OUTPUT PARAMETERS:

```
TYPE(Rn_GridComp1), INTENT(INOUT) :: gcRn   ! Grid Component
TYPE(Chem_Bundle), INTENT(INOUT) :: w_c ! Chemical tracer fields
```

INPUT PARAMETERS:

```
TYPE(ESMF_State), INTENT(inout) :: impChem    ! Import State
INTEGER, INTENT(IN) :: nymd, nhms       ! time
REAL,    INTENT(IN) :: cdt       ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(ESMF_State), intent(inout) :: expChem     ! Export State
INTEGER, INTENT(OUT) ::  rc                   ! Error return code:
                                              !  0 - all is well
                                              !  1 -
```

DESCRIPTION:
This routine implements the Rn driver. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
12Apr2008  Nielsen  Configured for radon
```

45.6 Rn_GridCompFinalize — The Chem Driver

INTERFACE:

```
SUBROUTINE Rn_GridCompFinalize1_ ( gcRn, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT/OUTPUT PARAMETERS:

```
TYPE(Rn_GridComp1), INTENT(INOUT) :: gcRn   ! Grid Component
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), INTENT(IN)  :: w_c      ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms       ! time
REAL,    INTENT(IN) :: cdt       ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(ESMF_State), INTENT(INOUT) :: impChem ! Import State
TYPE(ESMF_State), INTENT(INOUT) :: expChem ! Import State
INTEGER, INTENT(OUT) ::  rc                 ! Error return code:
                                            !  0 - all is well
                                            !  1 -
```

DESCRIPTION:
This routine finalizes this Grid Component. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

45.7 Rn_SingleInstance_ — Runs single instance of method

INTERFACE:

```
SUBROUTINE Rn_SingleInstance_ ( Method_, instance, &
                                gcRn, w_c, impChem, expChem, &
                                nymd, nhms, cdt, rc )
```

USES:

```
USE Rn_GridCompMod
USE ESMF
USE MAPL_Mod
USE Chem_Mod

IMPLICIT NONE
```

INPUT PARAMETERS:

```
 Input "function pointer"
 -----------------------
 INTERFACE
   SUBROUTINE Method_ (gc, w, imp, exp, ymd, hms, dt, rcode )
     USE Rn_GridCompMod
     USE ESMF
     USE MAPL_Mod
     USE Chem_Mod
     TYPE(Rn_GridComp1),  INTENT(INOUT)  :: gc
     TYPE(Chem_Bundle),   INTENT(IN)     :: w
     TYPE(ESMF_State),    INTENT(INOUT)  :: imp
     TYPE(ESMF_State),    INTENT(INOUT)  :: exp
     INTEGER,         INTENT(IN)     :: ymd, hms
     REAL,        INTENT(IN)     :: dt
     INTEGER,         INTENT(OUT)    :: rcode
   END SUBROUTINE Method_
 END INTERFACE

 INTEGER, INTENT(IN)           :: instance   ! instance number

 TYPE(Chem_Bundle), INTENT(INOUT) :: w_c     ! Chemical tracer fields
 INTEGER, INTENT(IN) :: nymd, nhms        ! time
 REAL,    INTENT(IN) :: cdt        ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(Rn_GridComp1), INTENT(INOUT) :: gcRn    ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem  ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem  ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Finalizes the Rn Grid Component. Multiple instance version. REVISION HISTORY:

```
27Feb2008  da Silva  Introduced multiple instances
12Apr2008  Nielsen   Configured for radon.
```

46 Module SS_GridCompMod — SS Grid Component Class

INTERFACE:

```
module  SSGridCompMod
```

USES:

```
USE ESMF
USE MAPL_Mod

use Chem_Mod              ! Chemistry Base Class
use Chem_StateMod         ! Chemistry State
use Chem_SettlingMod      ! Settling
use Chem_ConstMod, only: grav, von_karman, cpd     ! Constants !
use Chem_UtilMod          ! I/O
use Chem_MieMod           ! Aerosol LU Tables, calculator
use m_inpak90             ! Resource file management
use m_die, only: die
use m_mpout
use SeasaltEmissionMod
use DryDepositionMod
use WetRemovalMod
use ConvectionMod         ! Offline convective mixing/scavenging
```

PUBLIC TYPES:

```
PRIVATE
PUBLIC  SS_GridComp       ! The SS object
```

```
!PUBLIIC MEMBER FUNCTIONS:

 PUBLIC  SS_GridCompInitialize
 PUBLIC  SS_GridCompRun
 PUBLIC  SS_GridCompFinalize
```

DESCRIPTION:
This module implements the (pre-ESMF) SS Grid Component. REVISION HISTORY:

```
16Sep2003 da Silva  First crack.
```

46.1 SS_GridCompInitialize — Initialize SS_GridComp

INTERFACE:

```
subroutine SS_GridCompInitialize ( gcSS, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

INPUT PARAMETERS:

```
type(Chem_Bundle), intent(inout) :: w_c     ! Chemical tracer fields
integer, intent(in) :: nymd, nhms           ! time
real,    intent(in) :: cdt                  ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
type(SS_GridComp), intent(inout) :: gcSS     ! Grid Component
type(ESMF_State), intent(inout)  :: impChem  ! Import State
type(ESMF_State), intent(inout)  :: expChem  ! Export State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Initializes the SS Grid Component. It primarily sets the import state for each active constituent package. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

46.2 SS_GridCompRun — The Chem Driver

INTERFACE:

```
subroutine SS_GridCompRun ( gcSS, w_c, impChem, expChem, &
                            nymd, nhms, cdt, rc )
```

USES:

INPUT/OUTPUT PARAMETERS:

```
type(SS_GridComp), intent(inout) :: gcSS   ! Grid Component
type(Chem_Bundle), intent(inout) :: w_c      ! Chemical tracer fields
```

INPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: impChem    ! Import State
integer, intent(in) :: nymd, nhms          ! time
real,    intent(in) :: cdt                 ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: expChem     ! Export State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This routine implements the so-called SS Driver. That is, adds chemical tendencies to each of the constituents, Note: water wapor, the first constituent is not considered a chemical constituents. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

46.3 SS_Compute_Diags - Calculate seasalt 2D diagnostics

INTERFACE:

```
subroutine SS_Compute_Diags ( i1, i2, j1, j2, km, nbins, gcSS, w_c, tmpu, rhoa, u, v, &
                              sfcmass, colmass, mass, exttau, scatau, &
                              sfcmass25, colmass25, mass25, exttau25, scatau25, &
                              conc, extcoef, scacoef, exttaufm, scataufm, &
                              angstrom, fluxu, fluxv, rc )
```

USES:

INPUT PARAMETERS:

```
integer, intent(in) :: i1, i2, j1, j2, km, nbins
type(SS_GridComp), intent(inout):: gcSS     ! SS Grid Component
type(Chem_Bundle), intent(in)   :: w_c      ! Chem Bundle
real, pointer, dimension(:,:,:) :: tmpu     ! temperature [K]
real, pointer, dimension(:,:,:) :: rhoa     ! air density [kg m-3]
real, pointer, dimension(:,:,:) :: u        ! east-west wind [m s-1]
real, pointer, dimension(:,:,:) :: v        ! north-south wind [m s-1]
```

OUTPUT PARAMETERS:

```
type(Chem_Array), intent(inout)  :: sfcmass   ! sfc mass concentration kg/m3
type(Chem_Array), intent(inout)  :: colmass   ! col mass density kg/m2
type(Chem_Array), intent(inout)  :: mass      ! 3d mass mixing ratio kg/kg
type(Chem_Array), intent(inout)  :: exttau    ! ext. AOT at 550 nm
type(Chem_Array), intent(inout)  :: scatau    ! sct. AOT at 550 nm
type(Chem_Array), intent(inout)  :: sfcmass25 ! sfc mass concentration kg/m3 (pm2.5)
type(Chem_Array), intent(inout)  :: colmass25 ! col mass density kg/m2 (pm2.5)
type(Chem_Array), intent(inout)  :: mass25    ! 3d mass mixing ratio kg/kg (pm2.5)
type(Chem_Array), intent(inout)  :: exttau25  ! ext. AOT at 550 nm (pm2.5)
type(Chem_Array), intent(inout)  :: scatau25  ! sct. AOT at 550 nm (pm2.5)
type(Chem_Array), intent(inout)  :: conc      ! 3d mass concentration, kg/m3
type(Chem_Array), intent(inout)  :: extcoef   ! 3d ext. coefficient, 1/m
type(Chem_Array), intent(inout)  :: scacoef   ! 3d scat.coefficient, 1/m
type(Chem_Array), intent(inout)  :: exttaufm  ! fine mode (sub-micron) ext. AOT at 550 nm
type(Chem_Array), intent(inout)  :: scataufm  ! fine mode (sub-micron) sct. AOT at 550 nm
type(Chem_Array), intent(inout)  :: angstrom  ! 470-870 nm Angstrom parameter
type(Chem_Array), intent(inout)  :: fluxu     ! Column mass flux in x direction
type(Chem_Array), intent(inout)  :: fluxv     ! Column mass flux in y direction
integer, intent(out)             :: rc        ! Error return code:
                                              !  0 - all is well
                                              !  1 -
```

DESCRIPTION:
Calculates some simple 2d diagnostics from the SS fields Surface concentration (dry) Column mass load (dry) Extinction aot 550 (wet) Scattering aot 550 (wet) For the moment, this is hardwired. REVISION HISTORY:

```
16APR2004, Colarco
```

46.4 SS_Binwise_PM_Fractions - Calculate bin-wise PM fractions

INTERFACE:

```
subroutine SS_Binwise_PM_Fractions(fPM, rPM, r_low, r_up, nbins)
```

USES:

INPUT/OUTPUT PARAMETERS:

```
real, dimension(:), intent(inout) :: fPM     ! bin-wise PM fraction (r < rPM)
```

INPUT PARAMETERS:

```
real,    intent(in)              :: rPM     ! PM radius
integer, intent(in)              :: nbins   ! number of bins
real, dimension(:), intent(in)   :: r_low   ! bin radii - low bounds
real, dimension(:), intent(in)   :: r_up    ! bin radii - upper bounds
```

OUTPUT PARAMETERS:

46.5 SS_GridCompFinalize — The Chem Driver

INTERFACE:

```
subroutine SS_GridCompFinalize ( gcSS, w_c, impChem, expChem, &
                                 nymd, nhms, cdt, rc )
```

USES:

INPUT/OUTPUT PARAMETERS:

```
type(SS_GridComp), intent(inout) :: gcSS   ! Grid Component
```

INPUT PARAMETERS:

```
type(Chem_Bundle), intent(in)  :: w_c      ! Chemical tracer fields
integer, intent(in) :: nymd, nhms          ! time
real,    intent(in) :: cdt                 ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: impChem   ! Import State
type(ESMF_State), intent(inout) :: expChem   ! Import State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This routine finalizes this Grid Component. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

47 Module StratChem_GridCompMod - The StratChem Aerosol Grid Component

INTERFACE:

```
MODULE StratChemGridCompMod
```

USES:

```
USE ESMF
USE MAPL_Mod
USE Chem_Mod                              ! Chemistry Base Class
USE SC_GridCompMod                               ! ESMF parent component

IMPLICIT NONE
PRIVATE
```

PUBLIC MEMBER FUNCTIONS:

```
PUBLIC SetServices
```

DESCRIPTION:
`StratChem` is a ESMF gridded component implemented Code 613.3 Stratospheric Chemistry package. This code derives from the pre-ESMF SC component from GEOS-4. This GEOS-4 "component" used ESMF like constructs (Chem component class, import/export states, etc) but no ESMF specific data types because of an odd incompatibility with the fvGCM code (the so-called `oldworld` library. Unlike GEOS-4, here the Stratospheric Chemistry component is treated separately. REVISION HISTORY:

```
25feb2005  da Silva  First crack.
19jul2006  da Silva  First separate StratChem component.
```

47.1 SetServices — Sets IRF services for StratChem Grid Component

INTERFACE:

```
SUBROUTINE SetServices ( GC, RC )
```

ARGUMENTS:

```
 type(ESMF_GridComp), intent(INOUT) :: GC  ! gridded component
 integer, optional                  :: RC  ! return code
```

DESCRIPTION:
Sets Initialize, Run and Finalize services. REVISION HISTORY:

```
25feb2005  da Silva  First crack.
```

47.2 Initialize_ — Initialize StratChem

INTERFACE:

```
SUBROUTINE Initialize_ ( gc, impChem, expChem, clock, rc )
```

USES:

INPUT PARAMETERS:

```
type(ESMF_Clock),  intent(inout) :: clock      ! The clock
```

OUTPUT PARAMETERS:

```
type(ESMF_GridComp), intent(inout)  :: gc      ! Grid Component
type(ESMF_State), intent(inout) :: impChem     ! Import State
type(ESMF_State), intent(inout) :: expChem     ! Export State
integer, intent(out) ::  rc                    ! Error return code:
                                               !  0 - all is well
                                               !  1 -
```

DESCRIPTION:
This is a simple ESMF wrapper. REVISION HISTORY:

```
27Feb2005 da Silva  First crack.
```

47.3 Run_ — Runs StratChem

INTERFACE:

```
SUBROUTINE Run_ ( gc, impChem, expChem, clock, rc )
```

USES:

INPUT PARAMETERS:

```
type(ESMF_Clock),  intent(inout) :: clock      ! The clock
```

OUTPUT PARAMETERS:

```
type(ESMF_GridComp), intent(inout)  :: gc      ! Grid Component
type(ESMF_State), intent(inout) :: impChem     ! Import State
type(ESMF_State), intent(inout) :: expChem     ! Export State
integer, intent(out) ::  rc                    ! Error return code:
                                               !  0 - all is well
                                               !  1 -
```

DESCRIPTION:
This is a simple ESMF wrapper. REVISION HISTORY:

```
27Feb2005 da Silva  First crack.
```

47.4 Finalize_ — Finalize SC_GridComp (ESMF)

INTERFACE:

```
SUBROUTINE Finalize_ ( gc, impChem, expChem, clock, rc )
```

USES:

INPUT PARAMETERS:

```
type(ESMF_Clock),  intent(inout) :: clock      ! The clock
```

OUTPUT PARAMETERS:

```
type(ESMF_GridComp), intent(inout)  :: gc      ! Grid Component
type(ESMF_State), intent(inout) :: impChem     ! Import State
type(ESMF_State), intent(inout) :: expChem     ! Export State
integer, intent(out) ::  rc                    ! Error return code:
                                               !  0 - all is well
                                               !  1 -
```

DESCRIPTION:
This is a simple ESMF wrapper. REVISION HISTORY:

```
27Feb2005 da Silva  First crack.
```

48 Module SU_GridCompMod — SU Grid Component Class

INTERFACE:

```
module  SUGridCompMod
```

USES:

```
USE ESMF
USE MAPL_Mod

use Chem_Mod              ! Chemistry Base Class
use Chem_StateMod         ! Chemistry State
use Chem_ConstMod, only: grav, von_karman, cpd, &   ! Constants !
                         undefval  = > undef, airMolWght  = > airmw
use Chem_UtilMod          ! I/O
use Chem_MieMod           ! Aerosol LU Tables, calculator
use m_inpak90             ! Resource file management
use m_die, only: die
USE m_chars, ONLY: lowercase

use m_StrTemplate
use SulfateChemDriverMod
use ConvectionMod         ! Offline convective mixing/scavenging
use Chem_SettlingMod      ! Gravitiational Settling
```

PUBLIC TYPES:

```
PRIVATE
PUBLIC  SU_GridComp       ! The SU object
PUBLIC  SU_GridComp1      ! Single instance SU object
```

PUBLIC MEMBER FUNCTIONS:

```
PUBLIC  SU_GridCompInitialize
PUBLIC  SU_GridCompRun
PUBLIC  SU_GridCompFinalize
```

DESCRIPTION:
This module implements the (pre-ESMF) SU Grid Component. REVISION HISTORY:

```
16Sep2003 da Silva  First crack.
18May2006 da Silva  Removed ensure postive, now in GOCART_GridComp
25Aug2009 Nielsen   Connections, usage of GMI Combo OH, H2O2, and NO3
```

48.1 SU_GridCompInitialize — Initialize SU_GridComp

INTERFACE:

```
subroutine SU_GridCompInitialize ( gcSU, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

INPUT PARAMETERS:

```
type(Chem_Bundle), intent(inout) :: w_c        ! Chemical tracer fields
integer, intent(in) :: nymd, nhms           ! time
real, intent(in) :: cdt                     ! chemistry timestep (secs)
```

OUTPUT PARAMETERS:

```
type(SU_GridComp), intent(inout) :: gcSU   ! Grid Component
type(ESMF_State), intent(inout)  :: impChem  ! Import State
type(ESMF_State), intent(inout)  :: expChem  ! Export State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Initializes the SU Grid Component. It primarily sets the import state for each active constituent package. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

48.2 SU_GridCompRun — Run SU_GridComp

INTERFACE:

```
subroutine SU_GridCompRun ( gcSU, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), intent(in) :: w_c        ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms           ! time
REAL,    INTENT(IN) :: cdt                  ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(SU_GridComp), INTENT(INOUT) :: gcSU     ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem  ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem  ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Runs the CO Grid Component. Multiple instance version. REVISION HISTORY:

```
27Feb2008  da Silva  Introduced multiple instances
```

48.3 SU_GridCompFinalize — Initialize SU_GridComp

INTERFACE:

```
subroutine SU_GridCompFinalize ( gcSU, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

```
IMPLICIT NONE
```

INPUT PARAMETERS:

```
TYPE(Chem_Bundle), intent(in) :: w_c        ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms           ! time
REAL,    INTENT(IN) :: cdt                  ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(SU_GridComp), INTENT(INOUT) :: gcSU     ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem  ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem  ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Finalizes the SU Grid Component. Multiple instance version. REVISION HISTORY:

```
27Feb2008  da Silva  Introduced multiple instances
```

48.4 SU_GridCompInitialize — Initialize SU_GridComp

INTERFACE:

```
subroutine SU_GridCompInitialize1_ ( gcSU, w_c, impChem, expChem, &
                                     nymd, nhms, cdt, rc )
```

USES:

INPUT PARAMETERS:

```
type(Chem_Bundle), intent(inout) :: w_c        ! Chemical tracer fields
integer, intent(in) :: nymd, nhms           ! time
real, intent(in) :: cdt                     ! chemistry timestep (secs)
```

OUTPUT PARAMETERS:

```
type(SU_GridComp1), intent(inout) :: gcSU   ! Grid Component
type(ESMF_State), intent(inout)   :: impChem  ! Import State
type(ESMF_State), intent(inout)   :: expChem  ! Export State
integer, intent(out) ::  rc                   ! Error return code:
                                              !  0 - all is well
                                              !  1 -
```

DESCRIPTION:
Initializes the SU Grid Component. It primarily sets the import state for each active constituent package. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

48.5 SU_GridCompRun — The Chem Driver

INTERFACE:

```
subroutine SU_GridCompRun1_ ( gcSU, w_c, impChem, expChem, &
                              nymd, nhms, cdt, rc )
```

USES:

INPUT/OUTPUT PARAMETERS:

```
type(SU_GridComp1), intent(inout) :: gcSU   ! Grid Component
type(Chem_Bundle), intent(inout)  :: w_c    ! Chemical tracer fields
```

INPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: impChem    ! Import State
integer, intent(in) :: nymd, nhms          ! time
real, intent(in) :: cdt                    ! chemistry timestep (secs)
```

OUTPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: expChem   ! Export State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This routine implements the so-called SU Driver. That is, adds chemical tendencies to each of the constituents, Note: water wapor, the first constituent is not considered a chemical constituents. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

48.6 SU_ChemDrv - Do SU cycle chemistry following GOCART

INTERFACE:

```
subroutine SU_ChemDrv ( i1, i2, j1, j2, km, nbins, cdt, nymd, nhms, gcSU, &
                        w_c, ustar, u, v, shflux, oro, pblh, tmpu, &
                        cloud, rhoa, hghte, &
                        su_dep, &
                        su_pSO2, su_pMSA, su_pSO4g, su_pSO4aq, &   ! 2d diagnostics
                        pSO2, pMSA, pSO4g, pSO4aq,  &              ! 3d diagnostics
                        xoh, xno3, xh2o2, &
                        su_emis, &
                        rc)
```

USES:

INPUT PARAMETERS:

```
integer, intent(in) :: i1, i2, j1, j2, km, nbins, nymd, nhms
real, intent(in)    :: cdt
type(SU_GridComp1), intent(inout)   :: gcSU       ! SU Grid Component
real, pointer, dimension(:,:,:)     :: tmpu, cloud, rhoa, u, v, hghte
real, pointer, dimension(:,:)       :: ustar, shflux, oro, pblh
real, pointer, dimension(:,:,:)     :: xoh, xno3, xh2o2
```

OUTPUT PARAMETERS:

```
type(Chem_Bundle), intent(inout) :: w_c         ! Chemical tracer fields
type(Chem_Array), intent(inout)  :: su_dep(nbins), su_emis(nbins)  ! Mass lost by depositi
                                                  ! to surface, kg/m2/s
chemical production terms d(mixing ratio) /s
type(Chem_Array), intent(inout)  :: su_pSO2, su_pMSA, su_pSO4g, su_pSO4aq
type(Chem_Array), intent(inout)  :: pSO2, pMSA, pSO4g, pSO4aq

integer, intent(out)             :: rc          ! Error return code:
                                                !  0 - all is well
                                                !  1 -
character(len = *), parameter :: myname  =  'SU_ChemDrv'
```

DESCRIPTION:

Updates the SU concentration due to chemistry The SU grid component is currently established with 4 different species (bins) following this convection: 1) DMS 2) SO2 3) SO4 4) MSA Accordingly we have 4 chemical cycles to follow through, which are sub-subroutines under this one. The chemistry is a function of OH, NO3, and H2O2 concentrations as well as DMS, SO2, SO4, MSA concentrations. It is also a function of solar zenith angle and temperature. We pass in temperature. SZA will be a function of time of day and lat/lon. For now we simply add this to the grid component before calculating it. I bet this is somewhere else in the model. REVISION HISTORY:

```
06Nov2003, Colarco
```

48.7 SU_Wet_Removal - Removal of dust by precipitation

NOTE: For the removal term, fluxout is the sum of the in-cloud convective and large-scale washout and the total flux across the surface due to below-cloud (rainout) convective and large-scale precipitation reaching the surface. The fluxout is initialized to zero at the beginning and then at each i, j grid point it is added to. See Chin et al. 1996 for some of the logic of this. SO4 and MSA are scavenged "normally." DMS is not scavenged at all. SO2 is weakly soluble in water, but some fraction can be removed because of rapid aqueous phase reaction with H2O2. Accordingly, we compare the mixing ratios of H2O2 and SO2 and only scavenge that fraction of SO2 that is less than the H2O2 mixing ratio. If any of the scavenged SO2 is released by re-evaporation is emerges as SO4
INTERFACE:

```
subroutine SU_Wet_Removal ( i1, i2, j1, j2, km, nbins, cdt, rhoa, gcSU, w_c,&
                            precc, precl, dqcond, dqrl, tmpu, fluxout, pSO4wet_colflux, &
                            pso4wet, rc )
```

USES:

INPUT PARAMETERS:

```
integer, intent(in) :: i1, i2, j1, j2, km, nbins
real, intent(in)    :: cdt
real, pointer, dimension(:,:)   :: precc ! total convective precip [mm day-1]
real, pointer, dimension(:,:)   :: precl ! total large-scale prec. [mm day-1]
real, pointer, dimension(:,:,:) :: dqcond  ! change in q due to moist
                                           ! processes [kg kg-1 s-1]
real, pointer, dimension(:,:,:) :: dqrl    ! large-scale rainwater source [kg kg-1 s-1]
real, pointer, dimension(:,:,:) :: tmpu    ! temperature [K]
real, pointer, dimension(:,:,:) :: rhoa    ! air density [kg m-3]
```

OUTPUT PARAMETERS:

```
type(SU_GridComp1), intent(inout) :: gcSU  ! SU Grid Component
type(Chem_Bundle), intent(inout) :: w_c        ! Chemical tracer fields
type(Chem_Array), intent(inout)  :: fluxout(nbins) ! Mass lost by wet dep
```

```
                                                ! to surface, kg/m2/s
type(Chem_Array), intent(inout)  :: pSO4wet_colflux  ! aqueous chemical production of SO4 i
type(Chem_Array), intent(inout)  :: pSO4wet    ! aqueous chemical production of SO4 from SO
integer, intent(out)             :: rc         ! Error return code:
                                               !  0 - all is well
                                               !  1 -
character(len = *), parameter :: myname  =  'SU_Wet_Removal'
```

DESCRIPTION:
Updates the dust concentration in each vertical layer due to wet removal REVISION HISTORY:

```
17Nov2003, Colarco
```

48.8 SU_Compute_Diags - Calculate dust 2D diagnostics

INTERFACE:

```
subroutine SU_Compute_Diags ( i1, i2, j1, j2, km, nbins, gcSU, w_c, tmpu, rhoa, u, v, &
                              dmssfcmass, dmscolmass, so2sfcmass, &
                              so2colmass, so4sfcmass, so4colmass, &
                              exttau, scatau, so4mass, so4conc, extcoef, &
                              scacoef, angstrom, fluxu, fluxv, rc )
```

USES:

INPUT PARAMETERS:

```
integer, intent(in) :: i1, i2, j1, j2, km, nbins
type(SU_GridComp1), intent(inout):: gcSU     ! SU Grid Component
type(Chem_Bundle), intent(in)   :: w_c
real, pointer, dimension(:,:,:) :: tmpu      ! temperature [K]
real, pointer, dimension(:,:,:) :: rhoa      ! air density [kg m-3]
real, pointer, dimension(:,:,:) :: u         ! east-west wind [m s-1]
real, pointer, dimension(:,:,:) :: v         ! north-south wind [m s-1]
```

OUTPUT PARAMETERS:

```
type(Chem_Array), intent(inout)  :: dmssfcmass ! sfc mass concentration kg/m3
type(Chem_Array), intent(inout)  :: dmscolmass ! col mass density kg/m2
type(Chem_Array), intent(inout)  :: so2sfcmass ! sfc mass concentration kg/m3
type(Chem_Array), intent(inout)  :: so2colmass ! col mass density kg/m2
```

```
type(Chem_Array), intent(inout)  :: so4sfcmass ! sfc mass concentration kg/m3
type(Chem_Array), intent(inout)  :: so4colmass ! col mass density kg/m2
type(Chem_Array), intent(inout)  :: exttau     ! ext. AOT at 550 nm
type(Chem_Array), intent(inout)  :: scatau     ! sct. AOT at 550 nm
type(Chem_Array), intent(inout)  :: so4mass    ! 3D sulfate mass mr
type(Chem_Array), intent(inout)  :: so4conc    ! 3D mass concentration, kg/m3
type(Chem_Array), intent(inout)  :: extcoef    ! 3D ext. coefficient, 1/m
type(Chem_Array), intent(inout)  :: scacoef    ! 3D scat.coefficient, 1/m
type(Chem_Array), intent(inout)  :: angstrom   ! 470-870 nm Angstrom parameter
type(Chem_Array), intent(inout)  :: fluxu      ! Column mass flux in x direction
type(Chem_Array), intent(inout)  :: fluxv      ! Column mass flux in y direction
integer, intent(out)             :: rc         ! Error return code:
                                               !  0 - all is well
                                               !  1 -
```

DESCRIPTION:
Calculates some simple 2d diagnostics from the SU fields NOTE: For now this operates solely on the sulfate bin!!!! Surface concentration (dry) Column mass load (dry) Extinction aot 550 (wet) Scattering aot 550 (wet) For the moment, this is hardwired. REVISION HISTORY:

```
16APR2004, Colarco
```

48.9 SU_GridCompFinalize — The Chem Driver

INTERFACE:

```
subroutine SU_GridCompFinalize1_ ( gcSU, w_c, impChem, expChem, &
                                   nymd, nhms, cdt, rc )
```

USES:

INPUT/OUTPUT PARAMETERS:

```
type(SU_GridComp1), intent(inout) :: gcSU   ! Grid Component
```

INPUT PARAMETERS:

```
type(Chem_Bundle), intent(in)  :: w_c      ! Chemical tracer fields
integer, intent(in) :: nymd, nhms          ! time
real,    intent(in) :: cdt                 ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
type(ESMF_State), intent(inout) :: impChem   ! Import State
type(ESMF_State), intent(inout) :: expChem   ! Import State
integer, intent(out) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
This routine finalizes this Grid Component. REVISION HISTORY:

```
18Sep2003 da Silva  First crack.
```

48.10 SU_SingleInstance_ — Runs single instance of method

INTERFACE:

```
subroutine SU_SingleInstance_ ( Method_, instance, &
                                gcSU, w_c, impChem, expChem, &
                                nymd, nhms, cdt, rc )
```

USES:

```
Use SU_GridCompMod
Use ESMF
Use MAPL_Mod
Use Chem_Mod

IMPLICIT NONE
```

INPUT PARAMETERS:

```
Input "function pointer"
-----------------------
interface
  subroutine Method_ (gc, w, imp, exp, ymd, hms, dt, rcode )
    Use SU_GridCompMod
    Use ESMF
    Use MAPL_Mod
    Use Chem_Mod
    type(SU_GridComp1),  intent(inout)  :: gc
    type(Chem_Bundle),   intent(in)     :: w
    type(ESMF_State),    intent(inout)  :: imp
    type(ESMF_State),    intent(inout)  :: exp
```

```
    integer,            intent(in)     :: ymd, hms
    real,               intent(in)     :: dt
    integer,            intent(out)    :: rcode
  end subroutine Method_
end interface

integer, intent(in)           :: instance   ! instance number

TYPE(Chem_Bundle), intent(inout) :: w_c     ! Chemical tracer fields
INTEGER, INTENT(IN) :: nymd, nhms           ! time
REAL,    INTENT(IN) :: cdt                  ! chemical timestep (secs)
```

OUTPUT PARAMETERS:

```
TYPE(SU_GridComp1), INTENT(INOUT) :: gcSU    ! Grid Component
TYPE(ESMF_State), INTENT(INOUT)  :: impChem  ! Import State
TYPE(ESMF_State), INTENT(INOUT)  :: expChem  ! Export State
INTEGER, INTENT(OUT) ::  rc                  ! Error return code:
                                             !  0 - all is well
                                             !  1 -
```

DESCRIPTION:
Finalizes the CO Grid Component. Multiple instance version. REVISION HISTORY:

```
27Feb2008  da Silva  Introduced multiple instances
```