Held-Suarez Benchmark for FVdycore

Max J. Suarez and Atanas Trayanov
Global Modeling and Assimilation Office

May 25, 2007

Contents

1 FV_HeldSuarez_Main **2**

2 GEOS_AgcmSimpleGridCompMod **3**

2.1 SetServices 4

3 GEOS_HSGridCompMod **5**

3.1 SetServices 7

3.2 State Descriptions 8

3.3 Initialize 8

3.4 Run 9

4 FVdycore_GridCompMod **10**

4.1 SetServices 11

4.2 State Descriptions 12

4.3 Initialize 14

4.4 Run 15

4.5 RunAddIncs 15

4.6 Finalize 16

4.7 Coldstart 16

1 Program FV_HeldSuarez_Main

This is the "cap" or main program for the Held-Suarez benchmark of FVdycore. In this simple application, the cap is very thin, relying on MAPL_Cap for all of the work.

Typical of MAPL-based applications, this main program has two ESMF gridded components as its "children": an instance of the MAPL_History component, which manages the diagnostics, and the root of a single MAPL hierarchy, which implements the computations. Having this simple structure is one of the conditions for using MAPL_Cap.

The behavior of the application is controlled through three resource (or configuration) files. The MAPL_Cap opens configuration files for itself and its two children. These have the default names: CAP.rc, ROOT.rc, and HISTORY.rc. They must be present in the run directory at run time. The name of the CAP's own resource file is fixed as "CAP.rc", since this is the resource from which the application "boots-up." The other two may be renamed from file names specified in the CAP.rc.

Grids:
We will use the simplest grid creation strategy supported by MAPL_Cap: that the cap fully defines the grid and passes it to the entire hierarchy below it. In the FV_HeldSuarez application, all components run on a single grid, which must be of the FV type (i.e., a regular lat-lon grid with the poles at grid centers). This grid can be created automatically by the CAP, using parameters specified in the CAP.rc. In fact, the CAP resources default to this grid type. (See Resources description below.) Of course, the size of this grid must be consistent with the the data in the FV restart, if one is specified (see FV section in this document for more details).

Time:
The cap contains the time do loop. It creates the application's "universal" clock and advances it each time through the loop by an interval referred to as the application's "heartbeat." Before this loop, it initializes the entire hierarchy, including the history component, reading any necessary restarts. After the time loop is completed, it finalizes all components, writing any requested checkpoint files.

The clock may be advanced at either the top or bottom of the loop depending on the desired behavior of the children. This is controlled from the configuration. All of this functionality is implemented in the generic MAPL_Cap and can be controlled from the CAP.rc. The clock created by MAPL_Cap uses a Gregorian calendar.

Resources in the CAP.rc:

Name	Description	Units	Default
CF_FILE:	Name of ROOT's config file	none	"ROOT.rc"
CF_FILE:	Name of HISTORY's config file	none	'HISTORY.rc'
TICK_FIRST:	Determines when clock is advanced	1 or 0	none
BEG_YY:	Beginning year (integer)	year	1
BEG_MM:	Beginning month (integer 1-12)	month	1
BEG_DD:	Beginning day of month (integer 1-31)	day	1
BEG_H:	Beginning hour of day (integer 0-23)	hour	0
BEG_M:	Beginning minute (integer 0-59)	minute	0
BEG_S:	Beginning second (integer 0-59)	second	0
END_YY:	Ending year (integer)	year	1
END_MM:	Ending month (integer 1-12)	month	1
END_DD:	Ending day of month (integer 1-31)	day	1
END_H:	Ending hour of day (integer 0-23)	hour	0

END_M:	**Ending minute (integer 0-59)**	**minute**	**0**
END_S:	**Ending second (integer 0-59)**	**second**	**0**
RUN_DT:	**App Clock Interval (the Heartbeat)**	**seconds**	**none**
LATLON:	**1 -¿ regular lat-lon; 0 -¿ custom grid**	**0 or 1**	**1**
NX:	**Processing elements in 1st dimension**	**none**	**1**
NY:	**Processing elements in 2nd dimension**	**none**	**1**
IM_WORLD:	**Grid size in 1st dimension**	**none**	**none**
JM_WORLD:	**Grid size in 2nd dimension**	**none**	**none**
LM:	**Grid size in 3rd dimension**	**none**	**1**
GRIDNAME:	**Optional grid name**	**none**	**'APPGRID'**
IMS:	**gridpoints in each PE along 1st dimension**	**none**	**IMS**
JMS:	**gridpoints in each PE along 2nd dimension**	**none**	**JMS**
POLEEDGE:	**1-¿gridedge at pole; 0-¿gridpoint at pole**	**0 or 1**	**0**
LON0:	**Longituce of center of first gridbox**	**degrees**	**-90.**

USES:

```
use MAPL_Mod
use GEOS_AgcmSimpleGridCompMod, only:  ROOT_SetServices => SetServices
```

2 Module GEOS_AgcmSimpleGridCompMod

This gridded component (GC) combines the the GC that implements the Finite-Volume (FV) dynamics, with a simple physics component that implements the Held-Suarez benchmark forcing for testing dry dynamical cores.

USES:

```
use ESMF_Mod
use MAPL_Mod

use FVdycore_GridCompMod,   only:  DYN_SetServices => SetServices
use GEOS_hsGridCompMod,     only:  PHS_SetServices => SetServices
```

PUBLIC MEMBER FUNCTIONS:

```
public SetServices
```

2.1 SetServices

This is the only method of this component, since all of its registered methods can be defaulted to their MAPL generic versions. SetServices merely creates the children through MAPL and connects their Import-Export states. FV has some Imports that are not used in the Held-Suarez benchmark. Since these have proper defaults in FV, they can simply be terminated here.

INTERFACE:

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

ARGUMENTS:

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

CODE:

```
!=============================================================================
!
! ErrLog Variables

    character(len=ESMF_MAXSTR)              :: IAm
    integer                                 :: STATUS
    character(len=ESMF_MAXSTR)              :: COMP_NAME

! Locals

    integer :: I
    integer :: DYN
    integer :: PHS

!=============================================================================

! Begin...

! Get my name and set-up traceback handle
! ---------------------------------------

    Iam = 'SetServices'
    call ESMF_GridCompGet( GC, NAME=COMP_NAME, RC=STATUS )
    VERIFY_(STATUS)
    Iam = trim(COMP_NAME) // trim(Iam)

! Register children with MAPL and go down their SS hierarchy
! ----------------------------------------------------------

    DYN  = MAPL_AddChild(GC, NAME='FVDYNAMICS', SS=DYN_SetServices, RC=STATUS)
    VERIFY_(STATUS)
```

```
    PHS  = MAPL_AddChild(GC, NAME='HSPHYSICS',  SS=PHS_SetServices, RC=STATUS)
    VERIFY_(STATUS)

! Register connections between children
! -------------------------------------

    call MAPL_AddConnectivity ( GC,                                        &
         SHORT_NAME  = (/ 'DUDT', 'DVDT', 'DTDT' /),                       &
         SRC_ID      =  PHS,                                               &
         DST_ID      =  DYN,                                               &
                                                              RC=STATUS  )
    VERIFY_(STATUS)

    call MAPL_AddConnectivity ( GC,                                        &
         SRC_NAME = (/ 'U    ', 'V    ', 'T    ', 'PLE  ' /),             &
         DST_NAME = (/ 'U    ', 'V    ', 'TEMP ', 'PLE  ' /),             &
         SRC_ID   =  DYN,                                                  &
         DST_ID   =  PHS,                                                  &
                                                              RC=STATUS  )
    VERIFY_(STATUS)


! SetServices clean-up on the way back up through the hierarchy
!-------------------------------------------------------------

    call MAPL_TerminateImport(GC, SHORT_NAME = (/'PHIS ','DPEDT'/), &
         CHILD = DYN, RC=STATUS)
    VERIFY_(STATUS)

    call MAPL_GenericSetServices( GC, RC=STATUS )
    VERIFY_(STATUS)

    RETURN_(ESMF_SUCCESS)
  end subroutine SetServices
```

3 Module GEOS_HSGridCompMod

GEOS_HSGridCompMod implements the Held and Suarez (1994) benchmark forcing for global atmospheric dynamical cores. The Williamson, et al. (1998) stratospheric modifications to the benchmark are also implemented, and a mechanism is included to add a localized, gaussian heat source.

Scientific Basis:

The Held-Suarez forcing: The wind tendencies are given by

$$\frac{\partial \mathbf{v}}{\partial t} = \cdots - k_v(\sigma)\,\mathbf{v}$$

where

$$k_v(\sigma) = k_f f_1(\sigma)$$

and

$$f_1(\sigma) = \max\left[0, \frac{\sigma - \sigma_b}{1 - \sigma_b}\right]$$

The heating tendencies are given by:

$$\frac{\partial T}{\partial t} = \cdots - k_T(\phi,\sigma)\,[\,T - T_{eq}(\phi,p)\,]$$

$$T_{eq} = \max\left[T_{strat}, \left(\frac{p}{p_\circ}\right)^{\kappa}\left(T_0 - (\Delta T)_y\,\sin^2\phi - (\Delta\theta)_z \log\left(\frac{p}{p_\circ}\right)\cos^2\phi\right)\right]$$

where

$$k_T = k_a + (k_s - k_a)\cos^4\phi\, f_1(\sigma).$$

The parameter values suggested by Held and Suarez (1994) are:

$$\sigma_b = .7,$$
$$k_f = 1\ \text{day}^{-1},$$
$$k_a = \frac{1}{40\ \text{days}},$$
$$k_s = \frac{1}{4\ \text{days}},$$
$$(\Delta T)_y = 60\text{K}\ ,$$
$$(\Delta\theta)_z = 10\text{K}\ ,$$
$$T_{strat} = 200\text{K}\ ,$$
$$T_0 = 315\text{K}\ .$$

The Williamson Modification: Williamson's modification in the stratosphere simply alters T_{eq} at upper levels. For $p < p_D$, the constant HS value is replaced with the following distribution:

$$T_{eq} = T_{strat}\left[\min\left(1, \frac{p}{p_D}\right)^{\frac{R_d\gamma_D}{g}} + \min\left(1, \frac{p}{p_I}\right)^{\frac{R_d\gamma_I}{g}} - 1\right],$$

where T_{strat} is as in HS,

$$p_I = p_D - \frac{p_D - p(\sigma=0)}{2}(1 + \tanh(\alpha\frac{|\phi| - \phi_0}{(\delta\phi)_0})),$$

and defaults for the other parameters are $\gamma_I = -3.345 \times 10^{-3}$ K m^{-1}, $\gamma_D = 2.0 \times 10^{-3}$ K m^{-1}, $p_D = 100$ hPa, $\alpha = 2.65$, $(\delta\phi)_0 = 15°$, $\phi_0 = 60°$.

The Local Heat Source: A local heat form can also be included. It has the form

$$Q(\lambda,\phi) = Q_{max} e^{-\frac{1}{2}\ \frac{(\phi_h-\phi)^2}{(\delta\phi)_h^2} + \frac{(\lambda_h-\lambda)^2}{(\delta\lambda)_h^2}}\ \ h(p),$$

where λ is the longitude, p_s is the surface pressure,

$$h(p) = \begin{cases} 0 & p < p_1 \\ \sin(\pi\frac{p_s-p}{p_s-p_1}) & p_1 \le p < p_s, \end{cases}$$

Q_{max} is in K day^{-1}, with a default of zero, and defaults for the other parameters are $\phi_h = \lambda_h = 0$, $(\delta\phi)_h = 5°$, $(\delta\lambda)_h = 30°$, p_1 =0 hPa.

Code Implementation:

All HS parameters are optionally adjustable from the configuration, with published values being the defaults. Its parameters are also adjustable from the configuration and also default to published values. The Williamson modification can be disabled by choosing $p_D = 0$. Finally, we also include a mechanism for adding a localized, gaussian heat source whose strength and shape can be controlled from the configuration. This is zero by default.

References:

Held, I. M., and M. J. Suarez, 1994: A proposal for the intercomparison of the dynamical cores of atmospheric general circulation models. *Bulletin of the American Meteorological Society*, **75(10)**, 1825-1830.

Williamson, D.L., J. G. Olson, and B. A. Boville, 1998: A comparison of semi-Lagrangian and Eulerian tropical climate simulations. *Mon. Wea. Rev.*, **126**, 1001-1012

USES:

```
use ESMF_Mod
use MAPL_Mod
```

PUBLIC MEMBER FUNCTIONS:

```
public  SetServices
```

3.1 SetServices – Sets ESMF services for this component

This version uses the MAPL_GenericSetServices, which sets the Initialize and Finalize services, as well as allocating our instance of the MAPL_MetaComp and putting it in the gridded component (GC). The MAPL_MetaComp contains an ESMF state to use as an internal state. This routine describes the contents of this Internal state, as well as of the conventional Import and Export states by making call to MAPL.

INTERFACE:

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

ARGUMENTS:

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

3.2 State Descriptions

The component uses all three states (Import, Export and Internal). There is no Private (non-ESMF) Internal state. All three are managed by MAPL. The Internal state contains only invariant, horizontally dependent quantities that are set in Initialize, therefore it never needs checkpointing.

IMPORT STATE:

Short Name	Units	Dims	Vert Loc	Long Name
U	m s^{-1}	HorzVert	Center	eastward_wind
V	m s^{-1}	HorzVert	Center	northward_wind
TEMP	K	HorzVert	Center	air_temperature
PLE	Pa	HorzVert	Edge	edge_pressure

INTERNAL STATE:

Short Name	Units	Dims	Vert Loc	Long Name
SPHI2	1	HorzOnly	None	sine_latitude_squared
CPHI2	1	HorzOnly	None	cosine_latitude_squared
HFCN	1	HorzOnly	None	horizontal_structure_of_localized_heating
P_I	1	HorzOnly	None	Williamson_interface_pressure

EXPORT STATE:

Short Name	Units	Dims	Vert Loc	Long Name
DTDT	Pa K s^{-1}	HorzVert	Center	pressure_weighted_air_temperature_tendency
DUDT	m s^{-2}	HorzVert	Center	eastward_wind_tendency
DVDT	m s^{-2}	HorzVert	Center	northward_wind_tendency
T_EQ	K	HorzVert	Center	equilibrium_temperature
THEQ	K	HorzVert	Center	equilibrium_potential_temperature
TAUX	N m^{-2}	HorzOnly	None	eastward_surface_stress
TAUY	N m^{-2}	HorzOnly	None	northward_surface_stress
DISS	W m^{-2}	HorzOnly	None	frictional_dissipation

3.3 Initialize

Here we initialize the internal state, which in HS contains two-dimensional invariant arrays used in the forcing calculations. They are done here simply for economy. Initialize calls MAPL_GenericInitialize. If the import state needs to be restarted, MAPL will do it if a restart file is provided.

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
QMAX:	**Local heating**	**K day^{-1}**	**.000**
X0:	**Central longitude of local heating, λ_h**	**degrees**	**0.0**
Y0:	**Central latitude of local heating, ϕ_h**	**degrees**	**0.0**
DX:	**Longitudinal width of local heating, $(\delta\lambda)_h$**	**degrees**	**30.**
DY:	**Latitudinal width of local heating, $(\delta\phi)_h$**	**degrees**	**5.0**
AFAC:	**Williamson's α parameter**	**1**	**2.65/15.**
PHI0:	**Williamson's ϕ_o parameter**	**degrees**	**60.**

3.4 Run

The Run method of the HS Gridded Component. It computes tendencies for temperature and winds as described above, as well as a number of diagnostics. These are all optional Exports. The input winds and/or temperatures may be Friendly. All Friendlies are updated using a time step obtained from 'RUN_DT:' in the configuration. Both U and V must be Friendly for the wind to be updated. It is an error for one to be Friendly and not the other.

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
```

RESOURCES:

Name	Description	Units	Default
TAUA:	**H-S k_a^{-1} parameter**	**days**	**40.**
TAUS:	**H-S k_s^{-1} parameter**	**days**	**4.0**
TAUF:	**H-S k_f^{-1} parameter**	**days**	**1.0**
SIG1:	**H-S σ_b parameter**	**days**	**0.7**
T0:	**H-S T_o parameter**	**K**	**315.**
DELV1:	**H-S $(\Delta\theta)_z$ parameter**	**K**	**10.**
DELH:	**H-S $(\Delta T)_y$ parameter**	**K**	**60.**
TSTRT:	**H-S T_{strat} parameter**	**K**	**200.**
DISSQ:	**Controls addition of dissipation heat.**	**1 OR 0**	**1**
P1:	**Local heating p_1 parameter**	**Pa**	**0.0**
GAM_I:	**Williamson's γ_I parameter**	**K m^{-1}**	**-3.345e-3**
GAM_D:	**Williamson's γ_D parameter**	**K m^{-1}**	**.002**
P_D:	**Williamson's p_D parameter**	**Pa**	**1.E4**

4 Module FVdycore_GridCompMod

This module implements the Finite Volume Dynamical Core (FVdycore) as an ESMF/MAPL gridded component.

Scientific Description:

FVdycore is a hydrostatic, grid-point dynamical core on a sphere that uses finite-volume discretization techniques. A complete description may be found in: Lin, S.-J. 2004, A vertically Lagrangian Finite-Volume Dynamical Core for Global Models. *Mon. Wea. Rev.*, **132**, 2293-2307.

Code Implementation:

The FVdycore code consists of four sub-components,

- `cd_core`: The C/D-grid shallow water dynamics
- `te_map`: Vertical remapping algorithm
- `trac2d`: Tracer advection
- `benergy`: Energy balance

Grids and Decompositions The current version supports 2D domain decomposition, in which the Import, Export, and Internal states are decomposed in longitude and latitude.

Internally, the data is redistributed ("transposed") to a latitude height decomposition

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.

USES:

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

 use dynamics_vars, only : T_TRACERS, T_FVDYCORE_VARS, &
                           T_FVDYCORE_GRID, T_FVDYCORE_STATE
 use dynamics_vars, only : a2d3d, d2a3d, c2a3d
 use dynamics_vars, only : dynamics_init
 use dynamics_vars, only : dynamics_clean
 use fv_initstatemod
 use addincsmod, only: add_incs

This is the Pilgirm communications library.

 use parutilitiesmodule, only : gsize, gid, parinit
```

PUBLIC MEMBER FUNCTIONS:

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

4.1 SetServices

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
```

4.2 State Descriptions

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.

IMPORT STATE:

Short Name	Units	Dims	Vert Loc	Long Name
DUDT	m s^{-2}	HorzVert	Center	eastward_wind_tendency
DVDT	m s^{-2}	HorzVert	Center	northward_wind_tendency
DTDT	Pa K s^{-1}	HorzVert	Center	delta-p_weighted_temperature_tendency
DPEDT	Pa s^{-1}	HorzVert	Edge	edge_pressure_tendency
PHIS	m^2 sec^{-2}	HorzOnly	None	surface_geopotential_height
QTR	unknown	Center	advected_quantities	

EXPORT STATE:

Short Name	Units	Dims	Vert Loc	Long Name
KEDYN	Pa m s^{-1}	HorzOnly	None	vertically_integrated_kinetic_energy_tendency_across_dynamics
PEDYN	Pa m s^{-1}	HorzOnly	None	vertically_integrated_potential_energy_tendency_across_dynamic
TEDYN	Pa m s^{-1}	HorzOnly	None	mountain_work_tendency_across_dynamics
KECDCOR	Pa m s^{-1}	HorzOnly	None	vertically_integrated_kinetic_energy_tendency_across_cdcore
PECDCOR	Pa m s^{-1}	HorzOnly	None	vertically_integrated_potential_energy_tendency_across_cdcore
TECDCOR	Pa m s^{-1}	HorzOnly	None	mountain_work_tendency_across_cdcore
KETEMAP	Pa m s^{-1}	HorzOnly	None	vertically_integrated_kinetic_energy_tendency_across_temap
PETEMAP	Pa m s^{-1}	HorzOnly	None	vertically_integrated_potential_energy_tendency_across_temap
TETEMAP	Pa m s^{-1}	HorzOnly	None	mountain_work_tendency_across_temap
KEGEN	W m^{-2}	HorzOnly	None	vertically_integrated_generation_of_kinetic_energy
KEPHYS	Pa m s^{-1}	HorzOnly	None	vertically_integrated_kinetic_energy_tendency_across_physics
PEPHYS	Pa m s^{-1}	HorzOnly	None	vertically_integrated_potential_energy_tendency_across_physics
TEPHYS	Pa m s^{-1}	HorzOnly	None	mountain_work_tendency_across_physics
CONVKE	W m^{-2}	HorzOnly	Center	vertically_integrated_kinetic_energy_convergence
CONVCPT	W m^{-2}	HorzOnly	Center	vertically_integrated_enthalpy_convergence
CONVPHI	W m^{-2}	HorzOnly	Center	vertically_integrated_geopotential_convergence
U	m s^{-1}	HorzVert	Center	eastward_wind
V	m s^{-1}	HorzVert	Center	northward_wind
T	K	HorzVert	Center	air_temperature
PL	Pa	HorzVert	Center	mid_level_pressure
ZLE	m	HorzVert	Edge	edge_heights
ZL	m	HorzVert	Center	mid_layer_heights
S	m	HorzVert	Center	mid_layer_dry_static_energy
PLE	Pa	HorzVert	Edge	edge_pressure

TH	K	HorzVert	Center	potential_temperature
PLK	Pa^{κ}	HorzVert	Center	mid-layer_p_to_kappa
EXNER	1	HorzVert	Center	Exner_function
OMEGA	Pa sec^{-1}	HorzVert	Center	vertical_pressure_velocity
MFX	Pa m s^{-1}	HorzVert	Center	pressure_weighted_eastward_wind
MFY	Pa m s^{-1}	HorzVert	Center	pressure_weighted_northward_wind
MFZ	kg m^{-2} s^{-1}	HorzVert	Edge	vertical_mass_flux
MFX_A	Pa m s^{-1}	HorzVert	Center	zonal_mass_flux
MFY_A	Pa m s^{-1}	HorzVert	Center	meridional_mass_flux
PV	m^2 kg^{-1} sec^{-1}	HorzVert	Center	ertels_potential_vorticity
Q	1	HorzVert	Center	specific_humidity
DUDTDYN	m sec^{-2}	HorzVert	Center	tendency_of_eastward_wind_due_to_dynamics
DVDTDYN	m sec^{-2}	HorzVert	Center	tendency_of_northward_wind_due_to_dynamics
DTDTDYN	K sec^{-1}	HorzVert	Center	tendency_of_air_temperature_due_to_dynamics
DQVDTDYN	sec^{-1}	HorzVert	Center	tendency_of_specific_humidity_due_to_dynamics
PREF	Pa	VertOnly	Edge	reference_air_pressure
PS	Pa	HorzOnly	None	surface_pressure
TA	K	HorzOnly	None	surface_air_temperature
QA	1	HorzOnly	None	surface_specific_humidity
US	m s^{-1}	HorzOnly	None	surface_eastward_wind
VS	m s^{-1}	HorzOnly	None	surface_northward_wind
SPEED	m s^{-1}	HorzOnly	None	surface_wind_speed
DZ	m	HorzOnly	None	surface_layer_height
SLP	Pa	HorzOnly	None	sea_level_pressure
TROPP	Pa	HorzOnly	None	tropopause_pressure
TROPT	K	HorzOnly	None	tropopause_temperature
TROPQ	kg/kg	HorzOnly	None	tropopause_specific_humidity
DELP	Pa	HorzVert	Center	pressure_thickness
U_DGRID	m s^{-1}	HorzVert	Center	eastward_wind_on_native_D-Grid
V_DGRID	m s^{-1}	HorzVert	Center	northward_wind_on_native_D-Grid
TV	K	HorzVert	Center	air_virtual_temperature
THV	K/Pa**kappa	HorzVert	Center	scaled_virtual_potential_temperature
DDELPDTDYN	Pa sec^{-1}	HorzVert	Center	tendency_of_pressure_thickness_due_to_dynamics
UKE	J m^{-1} s^{-1}	HorzOnly	None	eastward_flux_of_atmospheric_kinetic_energy
VKE	J m^{-1} s^{-1}	HorzOnly	None	northward_flux_of_atmospheric_kinetic_energy
UCPT	J m^{-1} s^{-1}	HorzOnly	None	eastward_flux_of_atmospheric_enthalpy
VCPT	J m^{-1} s^{-1}	HorzOnly	None	northward_flux_of_atmospheric_enthalpy
UPHI	J m^{-1} s^{-1}	HorzOnly	None	eastward_flux_of_atmospheric_potential_energy
VPHI	J m^{-1} s^{-1}	HorzOnly	None	northward_flux_of_atmospheric_potential_energy
UQV	kg m^{-1} s^{-1}	HorzOnly	None	eastward_flux_of_atmospheric_water_vapor
VQV	kg m^{-1} s^{-1}	HorzOnly	None	northward_flux_of_atmospheric_water_vapor
UQL	kg m^{-1} s^{-1}	HorzOnly	None	eastward_flux_of_atmospheric_liquid_water
VQL	kg m^{-1} s^{-1}	HorzOnly	None	northward_flux_of_atmospheric_liquid_water
UQI	kg m^{-1} s^{-1}	HorzOnly	None	eastward_flux_of_atmospheric_ice
VQI	kg m^{-1} s^{-1}	HorzOnly	None	northward_flux_of_atmospheric_ice
DKE	W m^{-2}	HorzOnly	None	tendency_of_atmosphere_kinetic_energy_content_due_to_dynami
DCPT	W m^{-2}	HorzOnly	None	tendency_of_atmosphere_dry_energy_content_due_to_dynamics
DPET	W m^{-2}	HorzOnly	None	tendency_of_atmosphere_topographic_potential_energy_due_to_
WRKT	W m^{-2}	HorzOnly	None	work_done_by_atmosphere_at_top
DQV	kg m^{-2} s^{-1}	HorzOnly	None	tendency_of_atmosphere_water_vapor_content_due_to_dynamics
DQL	kg m^{-2} s^{-1}	HorzOnly	None	tendency_of_atmosphere_liquid_water_content_due_to_dynamics
DQI	kg m^{-2} s^{-1}	HorzOnly	None	tendency_of_atmosphere_ice_content_due_to_dynamics
CNV	W m^{-2}	HorzOnly	None	generation_of_atmosphere_kinetic_energy_content
U850	m s^{-1}	HorzOnly	None	eastward_wind_at_850_hPa
U500	m s^{-1}	HorzOnly	None	eastward_wind_at_500_hPa

U250	m s^{-1}	HorzOnly	None	eastward_wind_at_250_hPa
V850	m s^{-1}	HorzOnly	None	northward_wind_at_850_hPa
V500	m s^{-1}	HorzOnly	None	northward_wind_at_500_hPa
V250	m s^{-1}	HorzOnly	None	northward_wind_at_250_hPa
T850	K	HorzOnly	None	air_temperature_at_850_hPa
T500	K	HorzOnly	None	air_temperature_at_500_hPa
T250	K	HorzOnly	None	air_temperature_at_250_hPa
Q850	kg kg^{-1}	HorzOnly	None	specific_humidity_at_850_hPa
Q500	kg kg^{-1}	HorzOnly	None	specific_humidity_at_500_hPa
Q250	kg kg^{-1}	HorzOnly	None	specific_humidity_at_250_hPa
H850	m	HorzOnly	None	height_at_850_hPa
H500	m	HorzOnly	None	height_at_500_hPa
H250	m	HorzOnly	None	height_at_250_hPa
OMEGA500	Pa s^{-1}	HorzOnly	None	omega_at_500_hPa
U50M	m s^{-1}	HorzOnly	None	eastward_wind_at_50_meters
V50M	m s^{-1}	HorzOnly	None	northward_wind_at_50_meters

INTERNAL STATE:

Short Name	Units	Dims	Vert Loc	Long Name
AK	Pa	VertOnly	Edge	hybrid_sigma_pressure_a
BK	1	VertOnly	Edge	hybrid_sigma_pressure_b
U	m s^{-1}	HorzVert	Center	eastward_wind
V	m s^{-1}	HorzVert	Center	northward_wind
PT	K $Pa^{-\kappa}$	HorzVert	Center	scaled_potential_temperature
PE	Pa	HorzVert	Edge	air_pressure
PKZ	Pa^{κ}	HorzVert	Center	pressure_to_kappa

4.3 Initialize

Initialises the private internal state and calls MAPL_Generic initialize for the rest. It also sets the invariant Exports associated wioth the vertical grid. (p_{ref}, a_k and b_k).

INTERFACE:

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

ARGUMENTS:

```
type(ESMF_GridComp), intent(inout) :: gc       ! composite 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, intent(out), OPTIONAL     :: rc       ! Error code:
```

RESOURCES:

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

4.4 Run

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(in)    :: clock
integer, intent(out), optional     :: rc
```

4.5 RunAddIncs

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
```

4.6 Finalize

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
```

4.7 Coldstart

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.**