OLAM

Ocean – Land – Atmosphere Model

Version 5.3

INPUT NAMELIST PARAMETERS

(Special version for OLAM-SOIL)

Introduction

This document describes the input parameters for the Ocean-Land-Atmosphere Model (OLAM) version 5.3. These parameters are defined in a text file, named OLAMIN, which is read by OLAM each time execution begins. The parameter values are defined in OLAMIN by the user and specify nearly all aspects of how the model is to be set up and run. A sample OLAMIN file is provided with the model source code and contains a particular set of parameter values that specifies a relatively simple (generic) model run. The model user modifies OLAMIN with a text editor to provide a different set of instructions to the model.

All parameters in OLAMIN are read in FORTRAN namelist format. OLAMIN contains 1 namelist, which begins the identifier &OLAMIN and ends with the symbol /. Most parameters in OLAMIN are single-valued. A few, however, require multiple values. Most multi-valued parameters pertain to the capability of local mesh refinement in OLAM. The user may specify any number of geographic zones to be refined (doubled in horizontal resolution), and for each such zone, a separate value of the multi-valued namelist parameter must be provided. Similar multi-valued parameters pertain not to each refined area, but instead to each *refinement level*. The difference is that, for example, if two refinements are made in separate locations of the base global mesh, the refinements will be of the same doubled resolution, or as we say, of the same 'refinement level.' In this case, both refinement areas use the same value of this class of OLAMIN variable. On the other hand, if one of these refinements is made inside a previously refined area, the latter refinement is of the next 'higher level', and thus uses a separate value of the OLAMIN variable. Refined areas and refinement levels further described in the entry for NGRIDS below.

Syntax notes:

- Multiple values entered for a parameter must be separated by commas
- A comma is not required at the end of a data line unless it is followed by additional lines of the same data
- More than 80 characters may be used for data entry
- Comments are begun with the ! character and may be on the same line as data

All OLAMIN parameters are members of a FORTRAN derived data structure, which is declared to be of type 'NL'. This data type appears explicitly in OLAMIN. For example, parameter **RUNTYPE** appears in OLAMIN as **NL%RUNTYPE**. Throughout this document, the 'NL%' character string is omitted from the parameter descriptions.

The OLAMIN file contains brief comments to help guide user editing. The following

section in this document describes all of the OLAMIN parameters in greater detail. The parameters are listed in the same order as they appear in the OLAMIN file. Cross-referencing to related parameters provides additional information. Therefore, a complete read-through of this document is recommended. All parameters, including cross-references, are in bold-face format for easy identification.

OLAMIN namelist parameters

Parameter name	Description
EXPNME character	Character string used to label the output, 64 characters maximum. The user may enter a custom message to identify the simulation or run. The character string is written to the standard output file. NOTE: 'R' should be avoided as the first character entered for **EXPNME**; this character activates the "rotational" or vector-invariant form of the horizontal equation of motion, which in OLAM is experimental and not applicable to most situations.
TEST_CASE integer	Flag used to select special test cases that have been implemented. These include cases 2 and 5 of the shallow-water test case suite (Williamson et al. 1992). For these cases, set **TEST_CASE** = 2 or 5. Other test cases include those from the Dynamical Core Model Intercomparison Project (DCMIP) Summer Colloquia held at NCAR in 2008, 2012, and 2016; current implementation of these cases is mostly geared to the 2016 cases, but some cases from 2012 still function in OLAM. For normal OLAM simulations (i.e., those that are not among these test cases), set **TEST_CASE** = 0. The **TEST_CASE** flag will control a few logical decisions within the code. Separately, one must set OLAMIN parameters appropriately for each specific test cases. For the shallow water test cases, sample namelist files called OLAMIN_swtc2 and OLAMIN_swtc5 are provided.
RUNTYPE character	Specifies one of 5 ways to run the model. **RUNTYPE** = 'MAKEGRID', Make the atmosphere, land, and sea grid files. Do not perform model integration. *This is a required first step for running a simulation.*

A ‘MAKEGRID’ run first constructs the model atmospheric grid. User-defined specifications for **MDOMAIN, NGRIDS, DELTAX, NXP, NGRDLL, GRDRAD, GRDLAT,** and **GRDLON** define the horizontal structure of the grid. In contrast to previous versions of OLAM with triangular atmospheric grid cells, OLAM’s grid is now always composed of hexagons, plus a few pentagons and heptagons. The vertical structure of the atmospheric grid is specified by parameters **NZP**, **NDZ**, **HDZ**, and **DZ**, or by **ZZ** alone.

The ‘MAKEGRID’ run next fills topography based on user settings of the parameters **ITOPOFLG** and **TOPO_DATABASE.** Topographic height is defined both at the center and at the vertices of each hexagon, with the center value adjusted if necessary so that it is within the max/min range of heights at the vertices. Each of the six triangular sectors within a hexagonal cell, formed by one edge of the hexagon and radii from the end points of that edge to the hexagon center, is a topographic planar surface defined by the elevations of its three vertices. Each triangular planar surface is subdivided along any line of intersection between it and horizontal atmospheric grid levels (specifically, the horizontal surfaces at height zm that vertically separate scalar grid cells). Each subdivision (if any) of each planar triangular sector surface of each hexagonal grid cell then defines a surface grid cell of the combined land/sea grid of OLAM. Thus, specifying finer vertical spacing on the atmospheric grid leads to higher horizontal resolution of the land grid in the direction of the topographic gradient. Geometric information, including cell area and the atmospheric grid cell to which a surface cell is in contact are attached to each surface cell.

Next, OLAM determines properties of each cell (i.e., land vs. water, soil textural class, and vegetation class) either by reading geographic datasets or by user specification according to the parameters **IVEGFLG**, **ISOILFLG**, **VEG_DATABASE**, and **SOIL_DATABASE**. The surface cells are accordingly partitioned into land and sea groups. Optionally (see **ISEAGRID**), for any hexagon atmospheric cell that has only sea cells (no land cells) beneath it, all its sea cells may be re-combined into a single (hexagonal) sea cell.

Finally, the ‘MAKEGRID’ run type writes atmospheric, land, and sea grid information each to a separate file whose path and filenames are specified in **GRIDFILE**, **LANDUSEFILE** and

SEAFILE. The model run will stop immediately after these files are generated and will not proceed with integration.

RUNTYPE = 'INITIAL',
Perform model integration.
This run is designated to be the first of a simulation.

This means that a simulation is begun at model time zero and all atmospheric and soil prognostic variables are initialized according to the **INITIAL** parameter. *Note: An 'INITIAL' run reads land, sea, and grid files that were generated by prior 'MAKESFC' and 'MAKEGRID' runs.*

Note: A "simulation" is an entire integration of the atmospheric model from initial conditions to a final time. A "run" is an individual submission of the model code to the computer for execution. A simulation consists of one or more runs.

RUNTYPE = 'HISTORY',
Perform model integration. This run will be history restarted.

The atmospheric and land prognostic variables are initialized for this run by reading in from a history file that was written by OLAM on a previous run. The filename of this history file is specified in **HFILIN**.

The 'HISTORY' option is used when a simulation is carried out over a series of two or more runs. (see **IOUTPUT** and **HFILPREF**.)

RUNTYPE = 'HISTADDGRID',
Perform model integration. This run will be history restarted.

The atmospheric and land prognostic variables are initialized for this run by reading in from a history file that was written by OLAM on a previous run. The filename of this history file is specified in **HFILIN**.

At the time of the restart, the user is applying new local mesh refinements by increasing **NGRIDS** from the value it had in the previous run that generated the history file (from which the model will restart). In this operation, the configuration of all pre-existing grids, including size, location, and vertical and horizontal resolution, MUST NOT BE CHANGED. This

	means that **NZP, NDZ, HDZ, DZ, MDOMAIN, NXP** must remain the same, and that for grid numbers less than or equal to the previous value of **NGRIDS** (see **NGRIDS_OLD**), the values of **NGRDLL, GRDRAD, GRDLAT, and GRDLON** must remain the same. *The HISTADDGRID option is somewhat experimental and should be used with considerable caution, with an expectation of carefully checking model fields to make sure the restart occurred successfully. There are certain disadvantages to this operation, including an inevitable shock to parts of the numerical solution as land surface features suddenly change. Thus, it is recommended that this option should be avoided in other than exceptional circumstances.* **RUNTYPE** = 'PLOTONLY', Generate plots. Do not perform model integration. The atmospheric and soil prognostic variables are initialized from the first history file listed in **PLT_FILES**. 'PLOTONLY' generates plots from the history files that are listed in **PLT_FILES**. **NPLT** and the values following **FLDNAME** specify the plotting instructions. **NPLT_FILES** specifies the number of files from which to read and plot. When **RUNTYPE** = 'PLOTONLY', **FRQPLT** is ignored.
TIMEUNIT character	Units of time for the parameter **TIMMAX**. The allowable values of **TIMEUNIT** are: 's' for seconds, 'm' for minutes, 'h' for hours, and 'd' for days. This parameter allows numerical values for **TIMMAX** to be specified within convenient ranges for any simulation, which may range in duration from seconds to years.
TIMMAX real	Time during a simulation (units specified by **TIMEUNIT**) when the current run is to stop. A simulation is an entire integration of the atmospheric model from initial conditions to a final time. A run is an individual submission of the model code to the computer for execution. A simulation consists of one or more runs. **TIMMAX** is the total time elapsed from the beginning of a

	simulation, starting with the first run, and is cumulative over successive runs. At the beginning of a simulation, TIME8 (an internal model variable) is set to zero. TIME8 contains the elapsed time in seconds since the beginning of a simulation. TIME8 continues to increase with each successive run, that is, it does not reset to zero at the beginning of each run. When TIME8 = **TIMMAX,** the current run ends. To continue a simulation, one history restarts the simulation (**RUNTYPE** = 'HISTORY') with **TIMMAX** set to a larger value than on the previous run.
ITIME1 integer **IDATE1** integer **IMONTH1** integer **IYEAR1** integer	**ITIME1** (hhmm) is Coordinated Universal Time (UTC) in hours and minutes when the simulation begins **IDATE1** is the date of the month when the simulation begins. **IMONTH1** is the month of the year when the simulation begins. **IYEAR1** (yyyy) is the year when the simulation begins. These parameters specify when the simulation begins (TIME8 = 0), not when an individual run begins. (see **TIMMAX** for definitions). Therefore, the values for **ITIME1**, **IDATE1**, **IMONTH1**, and **IYEAR1** are the same for all runs within a simulation. OLAM uses **ITIME1**, **IDATE1**, **IMONTH1**, and **IYEAR1** to determine the proper solar declination angle and to coordinate the model clock with dates and times of various observational datasets.
NZP Integer	Specifies an upper bound on the number of vertical grid levels in the model. For the case where **NDZ** = 1, **NZP** is the actual number of vertical grid levels used. For the case where **NDZ** > 1, OLAM computes a number of vertical levels to use based on **NDZ**, **HDZ**, and **DZ**, and the actual number used is the lesser of that number and **NZP**. It is normally preferable to set **NZP** to a large number (such as 200) to prevent **NZP** from limiting the number of levels. Inside the code, OLAM defines the variable MZA to denote the actual number of vertical levels. OLAM uses a type-C grid stagger in the vertical (also in the

	horizontal), with vertical velocity at vertical index k displaced ½ grid level higher than scalar variables and horizontal velocity that have the same vertical index k. Vertical velocity at k = MZA is fixed at zero, representing a rigid upper lid as the top boundary condition, while k = MZA-1 is the highest level where vertical velocity is prognosed. Scalars and horizontal velocity are prognosed up to and including the k = MZA level. OLAM does not use terrain-following coordinates. Instead, model levels are horizontal and intersect topography, and cut cells are used to represent the topographic surface on the grid. In locations where topographic height is at its lowest (usually over the oceans, unless locations below sea level are explicitly represented; see **HDZ**), the lower surface coincides with the k = 1 vertical velocity level. The k = 1 level for scalars and horizontal velocity is therefore always below the surface and is never used. Hence, the number of levels available to represent the atmosphere is MZA – 1. OLAM is capable of running as a single-level model in which the rigid lid is at a height of z = **DZ**(2). For this case, MZA = **NZP** = 2, which is the minimum allowed value for MZA and **NZP**.
NDZ integer	The number of height and vertical grid spacing values that will be used of those specified in the **HDZ** and **DZ** arrays to generate the model vertical grid level heights and thicknesses. At least 2 (**HDZ,DZ**) pairs are required, one to specify the height and vertical grid spacing of the bottom of the atmospheric grid, and the other to specify the height and vertical grid spacing of the top. Additional (**HDZ,DZ**) pairs may be specified at intermediate heights in the atmosphere. Over the interval between consecutive specified (**HDZ,DZ**) pairs, a constant stretch ratio between consecutive model layers is applied, matching the **DZ** value and approximately matching the **HDZ** value at the bottom and top of the interval. **NDZ =** 1, special case: Use **ZZ** values instead of **HDZ** and **DZ** values. **NDZ >** 1, use **HDZ** and **DZ** values (maximum **NDZ** value is 10).

HDZ real array	The heights that correspond to the values specified in the **DZ** array. **HDZ** and **DZ** provide a very flexible and compact way to define the model vertical grid level heights and thickness without needing to explicitly specify each level. The first value specified for **HDZ** is the height in meters (relative to sea level) of the bottom of the atmospheric grid. It is most commonly set to 0., but this eliminates Earth's below-sea-level areas by setting their surface elevation to zero. If representation of below-sea-level areas is required in a simulation, the first value of **HDZ** should be set to the lowest surface elevation that is considered important. (Earth's lowest surface elevation is the Dead Sea, which is currently about minus 420 m.) The first value specified for **DZ** is the vertical grid spacing of the lowest model level. The second value of **HDZ** must be larger than the first, and specifies the height at which the second value of **DZ** applies. Each subsequent value of **HDZ** must be larger than the previous one, and it specifies the height of the corresponding **DZ** value. A maximum of 10 values may be specified for **HDZ** and for **DZ**; however, the number that will actually be used is equal to the value specified for **NDZ**. In other words, **HDZ(NDZ)** is the last member of the **HDZ** array that is used. The value of **HDZ(NDZ)** is also taken as the height (in meters) of the upper boundary of the OLAM domain, subject to minor adjustment as described below. The upper boundary of the OLAM domain should normally be at least 30000 m high so that pressure at the top is near zero. This is required because the upper boundary is a rigid lid and thus exerts a downward force per unit area on the atmosphere equal to the pressure at the top boundary. **HDZ(NDZ)** is most often set to around 35000 to 40000 m. Over the interval between consecutive specified (**HDZ,DZ**) pairs, a constant stretch ratio between consecutive model layers is applied, matching the **DZ** value and approximately matching the **HDZ** value at the bottom and top of the interval. For example, suppose that the first pair is (0.,50.) and the second pair is (2000.,200.), meaning that the vertical grid spacing at sea level is to be 50 m and the vertical grid spacing 2000 m above sea level is to be 200 m. Given these specifications, OLAM will generate 19 vertical grid levels from sea level to approximately 2000 m, with

	level thicknesses of 50.0, 54.0, 58.3, 63.0, …, 185.2, and 200.0 m, which corresponds to a constant stretch ratio of 1.08 between consecutive levels. (The actual height of the top of the 19th grid level in this case is 2073.6 m, because it is not possible in general to exactly match **HDZ** values given the other constraints.) Use of a constant stretch ratio (i.e., geometric stretching of model levels) minimizes the maximum expansion ratio between consecutive levels for a given net stretching over a given number of vertical levels. It is best to keep the ratio between consecutive levels within 10% or 15% of unity. Large ratios degrade the accuracy of the vertical differencing in the model. The user should always examine the vertical grid that is generated by this method to make sure that no vertical stretch ratios are too high or low. The vertical grid information, including stretch ratios, are listed in the standard output file for every OLAM run In the special case where **NDZ** = 1, **HDZ** and **DZ** values are not used.
DZ real array	The vertical grid spacing, in meters, at selected heights (which are specified in **HDZ**) in the atmosphere. (See **NDZ** and **HDZ**.)
ZZ real array	Heights of grid levels, in meters relative to sea level, beginning at the lowest model level and continuing to the top model level. **NDZ** must be set to 1 for the model to use vertical grid levels from **ZZ**. The number of values specified for **ZZ** must be at least as large as **NZP**. Using **ZZ** instead of using **HDZ** and **DZ** is an alternative means of setting the vertical grid structure. Using **ZZ** allows one to explicitly specify the height of each model level. However, with **ZZ**, care must to be taken not to cause two consecutive levels to have grossly different vertical thicknesses, such as a factor of 2, because this degrades the accuracy of the model's vertical differencing schemes.
MDOMAIN integer	Specifies the whether the model horizontal domain is global or limited-area, and it also specified the general shape of the domain

	for the cases where it is limited area. In the current version of OLAM, three **MDOMAIN** options are currently functional. Several other options were implemented and tested in earlier OLAM versions. Some of these will be restored and others will be added to futures versions of OLAM. **MDOMAIN** = 0, Sets up a global domain over the spherical earth. The horizontal grid spacing is determined from the size of the earth and **NXP**. Construction of the OLAM grid begins by projecting the Earth's surface onto an inscribed icosahedron, which contains 20 triangles. Grid 1, which covers the globe, is defined by subdividing each triangle of the icosahedron into **NXP** * **NXP** smaller triangles. The triangular mesh is then adjusted for optimum grid cell size and shape. As a final step, the triangular mesh is exchanged for its dual mesh of hexagons. The resultant horizontal grid spacing of the dual hexagonal grid, defined to be the square root of the area of a grid cell, is approximately equal to7200 km / **NXP**. For example, to obtain a horizontal grid spacing of 100 km on grid, **NXP** should be set to 72. **MDOMAIN** = 3 (not currently functional), Sets up a limited area, 2D vertical cross section of channel flow between parallel walls, with cyclic boundary conditions at both ends of the channel. <u>The triangular mesh was previously used for this option; a replacement is planned for the hexagonal mesh.</u> The effective horizontal grid spacing is specified in **DELTAX,** and the number of points spanning the x direction is specified in **NXP**. **MDOMAIN** = 4 Sets up a limited area, 3D Cartesian domain, with cyclic lateral boundary conditions in both horizontal directions. In the y direction, the domain is fixed at only four grid points wide. The effective horizontal grid spacing is specified in **DELTAX,** and the number of points spanning the x direction is specified in **NXP**. **MDOMAIN** = 5 Sets up a limited area, 3D Cartesian domain that has the shape of a perfect hexagon, with cyclic lateral boundary conditions applied across all 3 sets of opposite edges. Thus, the hexagonal domain

	behaves as if an identical domain lies adjacent across each of its 6 edges. The hexagonal mesh is used. The effective horizontal grid spacing is specified in **DELTAX,** and the number of points spanning the radial distance of the domain is specified in **NXP**.
NXP integer	When **MDOMAIN** = 0, **NXP** is the number of grid points spanning one side of any triangle of the global icosahedrons that is constructed as the first step of generating OLAM's global grid. When **MDOMAIN** = 4, **NXP** is the number of grid points spanning the x direction. When **MDOMAIN** = 5, **NXP** is the number of grid points spanning the radial distance from the center to an edge of the hexagonal domain. For a global simulation domain [**MDOMAIN** = 0], the horizontal grid spacing for grid 1, defined to be the square root of the area of a grid cell, is determined from the size of the earth and **NXP**, and is approximately equal to7200 km / **NXP**. For example, to obtain a horizontal grid spacing of 100 km on grid, **NXP** should be set to 72. For a limited-area simulation [**MDOMAIN** = 4 or 5], the product **NXP** * **DELTAX** is the extent in meters of the full domain in the x direction, or in the case of the hexagonal domain, its radius. Note: In many cases where local mesh refinement is used (i.e., when **NGRIDS** > 1), **NXP** must be a multiple of 3. It is thus a good practice to routinely follow this rule, which is not particularly restrictive.
DELTAX real	Used only for a limited-area domain specification [**MDOMAIN** = 3, 4, or 5]. **DELTAX** is the horizontal grid spacing or grid cell size of the model coarse grid (grid 1) in units of meters. The product of **DELTAX** * **NXP** is the extent, in meters, of the full domain in the x direction. All spacings on the finer nested grids (grids with MRL >1) are derived from this value of **DELTAX** and the rule that grid spacing

	is halved for each unit increase in MRL. For a global simulation domain [**MDOMAIN** = 0], **DELTAX** is ignored and the horizontal grid spacing for grid 1 is computed from the size of the earth and **NXP**.
NGRIDS integer	Specifies the number of "grids" in the model run, including the "coarse grid" and any refined areas or "nested grids". The "coarse grid" is a mesh of nearly uniform resolution that covers the entire model simulation domain. A "nested grid" is a region of higher spatial resolution that is constructed in a limited area of its parent lower resolution grid. In OLAM, a nested grid always has double the resolution (half the horizontal grid spacing) of its parent grid. **NGRIDS** = 1, activates a single grid (grid 1) covering the entire spatial domain. **NGRIDS** = 2, activates grid 1 and one nested grid (grid 2) within grid 1. **NGRIDS** = 3, activates grids 1, 2, and 3. Grid 3 may be nested either in grid 1 or in grid 2. **NGRIDS** = 4, activates grids 1, 2, 3, and 4; and so on. Any nested grid may be nested within any other grid of lower number That is, the immediate parent may be any grid of lower number. The size, location, and orientation of each nested grid are specified by the parameters **NGRDLL, GRDRAD, GRDLAT,** and **GRDLON**. These 4 parameters automatically determine the parentage of each grid. No other specification (such as the parameter NXTNEST used in RAMS) is used in OLAM. The concept of "multiple grids" is used in OLAM only during initialization to establish the grid spacing at each geographic location. After grid initialization, the entire model domain is treated as having a single grid with spatially-variable resolution. Horizontal communication of field values between regions of different resolution is performed in exactly the same manner as within a region of uniform resolution, for example by means of the horizontal advection operator. OLAM does not use special nested grid communication computations as done in RAMS.

	The "mesh refinement level" (MRL) is an integer that defines how refined any portion of the (single) OLAM grid is relative to the coarsest regions of the grid, that is, relative to the resolution specified for grid 1. Grid 1 always has MRL = 1. Grid 2 always has MRL = 2. That is, grid 2 is always nested within grid 1 and therefore grid 2 has twice the resolution of grid 1. We thus say that grid 2 is of the next higher MRL than grid 1. Grid 3 will have MRL = 2 if its immediate parent is grid 1, or, grid 3 will have MRL = 3 if its immediate parent is grid 2.
NGRIDS_OLD integer	Uses the same concept of grids as **NGRIDS**, and specifies the value that **NGRIDS** had previously (on the history file used for history restarting) in the special case when **RUNTYPE** is set to 'HISTADDGRID'. For any other value of **RUNTYPE**, set **NGRIDS_OLD** = 0.
NGRDLL integer array **GRDRAD** real array **GRDLAT** real array **GRDLON** real array	**NGRDLL** is a one-dimensional array that specifies the number of geographic points that define the location and extent of each refined area of the grid. The user specifies a sequence of one or more points for each refined area, providing the number of points specified (**NGRDLL**), latitude and longitude of each point (**GRDLAT** and **GRDLON**), and a radius of influence in meters (**GRDRAD**). For each specified refinement area or "grid", OLAM performs a doubling of resolution over the geographic region that is within a distance of **GRDRAD** from any of the points specified for that grid or from the line segments connecting any two consecutive points in the sequence. It is permitted to specify only one point for a grid, in which case the refined area will be roughly circular in shape. **NGRDLL** must be defined for all "grids" from 2 up to and including **NGRIDS**. If **NGRIDS**=1, **NGRDLL** is not used. **GRDRAD** is a radius of influence to be used in defining a geographic region over which the grid is to be refined. It is used in combination with **NGRDLL**, **GRDLAT**, and **GRDLON**. **GRDRAD** must be defined for all "grids" from 2 up to and including **NGRIDS**. If **NGRIDS**=1, **GRDRAD** is not used. **GRDLAT** and **GRDLON** are two-dimensional arrays that specifiy the latitudes and longitudes of geographic points that define the location and extent of one or more refined areas of the grid ("nested grids"). The second index of these arrays is the grid number. It must range from 2 (since grid 1 refers to the base global grid, which is not a refined area) up to and including the

	value of **NGRIDS**. If **NGRIDS**=1, no values for **GRDLAT** and **GRDLON** are required. The first index of these arrays is the sequence number (first, second, etc.) of the geographic point specified for a given grid number. It must range from 1 up to and including **NGRDLL**(NG), which is the number of geographic points specified for grid number NG. For example, if we want to use one geographic point to define grid 2 and 2 geographic points to define grid 3, then we must define values for **GRDLAT**(1,2), **GRDLAT**(1,2), and **GRDLAT**(2,3), and the same for **GRDLON**. We must also set **NGRDLL**(2) = 1 and **NGRDLL**(3) = 2.
NSFCGRID_ROOT integer	Starting point in the construction of the atmospheric grid from which local refinements of the surface grid will proceed. **NSFCGRID_ROOT** must be less than or equal to **NGRIDS**. OLAM constructs its atmospheric grid first, beginning from the global grid, designated as "grid 1" (see **NXP**), and then applying any specified local refinements from "grid 2" to "grid **NGRIDS**." When this procedure completes the grid number that equals the value specified for **NSFCGRID_ROOT**, a copy is made of the atmospheric grid and set aside as a starting point for constructing the surface grid. Any additional local refinements of the atmospheric grid up to **NGRIDS** are then completed. Then, beginning from the copied preliminary surface grid, local mesh refinements (if any are specified in **NSFCGRIDS**) are applied as specified in **NSFCGRDLL**, **SFCGRDRAD**, **SFCGRDLAT**, and **SFCGRDLON**.
NSFCGRIDS integer	Total number of surface grid local mesh refinements that will be applied *independently* of those applied to the atmosphere grid. These refinements are the means by which the surface grid can be constructed at much higher resolution than the atmospheric grid. Details of the surface grid refinements are specified in **NSFCGRDLL**, **SFCGRDRAD**, **SFCGRDLAT**, and **SFCGRDLON**. Local refinements of the surface grid begin from a copy of the atmospheric grid that is made either at an intermediate step of its construction, in which case **NSFCGRID_ROOT** is less than **NGRIDS**, or after the final step of its construction, in which case **NSFCGRID_ROOT** equals **NGRIDS**. If **NSFCGRIDS** is set to zero, then no independent refinements of the surface grid are made, and the surface grid then is identical in horizontal footprint to the atmospheric grid (except for subsequent partitioning of surface grid cells in areas of sloping topography).

NSFCGRDLL integer array **SFCGRDRAD** real array **SFCGRDLAT** real array **SFCGRDLON** real array	**NSFCGRDLL** is a one-dimensional array that specifies the number of geographic points that define the location and extent of each refined area of the surface grid. The user specifies a sequence of one or more points for each refined area by providing the number of points specified (**NSFCGRDLL**), latitude and longitude of each point (**SFCGRDLAT** and **SFCGRDLON**), and a radius of influence in meters (**SFCGRDRAD**). For each specified refinement area or "grid", OLAM performs a doubling of resolution over the geographic region that is within a distance of **SFCGRDRAD** from any of the points specified for that grid or from the line segments connecting any two consecutive points in the sequence. It is permitted to specify only one point for a grid, in which case the refined area will be roughly circular in shape. **NSFCGRDLL** must be defined for all "grids" from 1 up to and including **NSFCGRIDS**. If **NSFCGRIDS**=0, **NSFCGRDLL** is not used. **SFCGRDRAD** is a radius of influence to be used in defining a geographic region over which the surface grid is to be refined. It is used in combination with **NSFCGRDLL**, **SFCGRDLAT**, and **SFCGRDLON**. **SFCGRDRAD** must be defined for all "grids" from 1 up to and including **NSFCGRIDS**. If **NSFCGRIDS**=0, **SFCGRDRAD** is not used. **SFCGRDLAT** and **SFCGRDLON** are two-dimensional arrays that specifiy the latitudes and longitudes of geographic points that define the location and extent of one or more refined areas of the surface grid ("nested grids"). The second index of these arrays is the grid number. It must range from 1 up to and including the value of **NSFCGRIDS**. If **NSFCGRIDS**=0, no values for **SFCGRDLAT** and **SFCGRDLON** are required. The first index of these arrays is the sequence number (first, second, etc.) of the geographic point specified for a given grid number. It must range from 1 up to and including **NSFCGRDLL**(NG), which is the number of geographic points specified for grid number NG. For example, if we want to use one geographic point to define grid 2 and 2 geographic points to define grid 3, then we must define values for **SFCGRDLAT**(1,2), **SFCGRDLAT**(1,2), and **SFCGRDLAT**(2,3), and the same for **SFCGRDLON**. We must also set **NSFCGRDLL**(2) = 1 and **NSFCGRDLL**(3) = 2.
DTLONG real	Length of the timestep, in seconds, at which most processes on the coarsest grid (grid 1, MRL = 1) will be updated. This is the primary model timestep to which all other timesteps relate.

	DTLONG should be set reasonably close to, but below the limiting value for computational stability of the model. Considering the full model equation set, this limiting value is determined by a number of factors, including wind velocity, internal gravity wave speed, external gravity wave speed (related in part to the height of the model domain), vertical and horizontal grid spacing, and maximum terrain slope. It is not easy to provide a comprehensive formula for estimating an optimal value for **DTLONG**, and the user should generally try a few values when beginning a new simulation to find a suitable **DTLONG** value. However, some guidelines can be provided based on experience. For a deep model domain, 20 km or more, **DTLONG** can usually be 5 or 6 seconds for a 1 km horizontal spacing; 30 seconds for a 10 km spacing; 90 seconds for a 60 km spacing; and 180 seconds or more for spacings over 150 km.
NACOUST integer array	MRL-dependent parameter. Specifies the number of short (acoustic) timesteps performed for each long timestep on each MRL. For the current version of OLAM, **NACOUST** must be the same for all MRLs. The short timesteps are used to evaluate the pressure gradient force and divergence equation terms and apply them to the velocity and pressure fields. That is, the short timesteps evaluate the terms involving the propagation of sound waves. The long timestep is used to evaluate most other terms, including diffusion, Coriolis force, microphysical tendencies, and scalar advection. This practice of time splitting of the prognostic equations allows full explicit computation, on the short timestep, of the terms governing the rapidly moving sound waves, while the terms governing slower processes are performed on the long timestep. **NACOUST** should be set to a value that keeps the length of an acoustic timestep on any grid to less than about 60 seconds. For example, if **DTLONG** is set to 180, **NACOUST** should be set to no less than 3. On any grid, the length of the acoustic timestep must be small enough that sound cannot propagate across a grid cell in a single timestep. For higher resolution grids, this will require the acoustic timestep to be smaller than 60 seconds.

INITIAL integer	Specifies how to initialize the model atmospheric fields. Used when **RUNTYPE** = 'INITIAL' or 'HISTORY'. Ignored when **RUNTYPE** = 'MAKESFC' or 'PLOTONLY'. **INITIAL** = 1, The model atmospheric fields are initialized horizontally homogeneously from a single sounding that is supplied by the user in parameters **PS**, **TS**, **RTS**, **US**, **VS**. This option is typically used when the model domain is a limited size (a few hundred kilometers across or less), although it may also be used for the global domain. **INITIAL** = 2, The model atmospheric fields are initialized from a global gridded pressure level file (see **IAPR**) that contains observational data. If the model top extends above the pressure level data, **ZONCLIM** specifies the file used to initialize the atmosphere above the pressure level data. **INITIAL** = 3, The model atmospheric fields are initialized by interpolation from a longitudinally homogenous zonal wind and temperature dataset. This dataset is contained in the file ZONAVG_CLIMATE, located in the /etc directory. The filename and location of this dataset is specified in **ZONCLIM**.
ZONCLIM character	Specifies the filename (with directory path) of a climatological dataset of monthly mean longitudinally homogeneous zonal wind and temperature. If **INITIAL** = 3, this file is read and used to initialize the atmosphere. If **INITIAL** = 2 and the model top extends above the pressure level data, this file is used to initialize the atmosphere above the pressure level data.
NUDFLAG integer	Flag that specifies whether or not to activate 4DDA (4 Dimensional Data Assimilation) by Newtonian nudging. Used only when **INITIAL** = 2. **NUDFLAG** = 0, no nudging. **NUDFLAG** = 1, nudge from pressure level files. Pressure level files are used to input time-dependent observed momentum, pressure-level height, temperature, and water vapor fields for 4DDA. The 4DDA option nudges the model

	fields toward the observed data during time integration The assimilation method is Newtonian nudging and is usually applied over the entire computational domain. However, a low-pass filter is usually first applied to both model and observed fields so that only larger scales will contribute to the computation of differences between model and observation. The model fields are nudged only to these large-scale differences, which results in nearly zero impact on smaller scale structures in the model (see **NUDNXP** and **TNUDCENT**). Because OLAM is usually run as a global model, it has no lateral boundaries and thus does not require time-dependent observational data at its lateral boundaries, as does RAMS. However, the **NUDFLAG** option is available for some applications where OLAM is run for several days or longer and it is desired that the large-scale fields in OLAM agree reasonably closely to observations over a specific time period.
MAX_NUD_MRL integer	If nudging is performed, this parameter provides an easy way to selectively nudge only coarser regions of the grid, if desired. **MAX_NUD_MRL** specifies the highest mesh refinement level (MRL) grid cells that will be nudged.
NUDNXP integer	Used only when **INITIAL** = 2 and **NUDFLAG** = 1 (4DDA activated). **NUDNXP** controls the cutoff scale of a low-pass filter that is applied to differences between modeled and observed state variables as part of the process of Newtonian-nudging of prognostic OLAM fields to the observations. Both model and observed variables are interpolated from the OLAM grid to an auxiliary quasi-uniform global grid of mostly hexagonal (plus 12 pentagonal) cells, and the differences between model values and observations are evaluated on that grid. Nudging tendencies computed from these differences and from **TNUDCENT** are interpolated back to the OLAM grid and applied to the prognostic model fields. **NUDNXP** is used to define the resolution of this auxiliary grid in the same way that **NXP** is used to define the main OLAM grid (prior to any local mesh refinement). **NUDNXP** should normally be less than **NXP** in order to avoid nudging the finest scales that are present on the OLAM grid. **NUDNXP** should also be chosen according to the scales of disturbances in the observed data that one wants to force into the OLAM

	prognostic fields. The smallest length scale that is represented on the auxiliary grid (i.e., as a 2-delta-x feature) is approximately 14,000 km / **NUDNXP**. Thus, for example, setting **NUDNXP** to 35 will generate an auxiliary grid that resolves scales down to about 400 km. Such a grid would capture most synoptic scale and larger disturbances, while filtering out most mesoscale features. Special case: **NUDNXP** = 0 (when **NUDFLAG** = 1) indicates that nudging will be point-by-point. That is, a scale filter will NOT be applied, and nudging is applied directly at each grid point based on the difference between model and observed values interpolated to that point.
TNUDCENT real	Used only when **INITIAL** = 2 and **NUDFLAG** = 1 (4DDA activated). **TNUDCENT** controls the strength of the Newtonian nudging used in 4DDA. The nudging source term in the tendency equation is given by (I - M)/**TNUDCENT**, where I is a filtered pressure level data value at a particular location and M is the corresponding filtered model value. Thus, TNUDCENT is the e-folding time of the difference between I and M. Because I and M are low-pass filtered values, only the longer scales are nudged. This enables smaller-scale structures to freely develop in the OLAM fields without suffering damping due to the nudging. The value of **TNUDENT** should correspond roughly to characteristic time scales of the length scales being nudged. This is usually at least a few hours. (see **NUDFLAG** and **NUDNXP**)
IOUTPUT integer	Flag that controls whether or not the model will output history files. **IOUTPUT** = 0, no output files will be generated. **IOUTPUT** = 1, the model will generate output files in HDF5 format. OLAM adds a special character (H) to history file names to distinguish them from monthly-average and daily-average history files (see **HFILEPREF**, **ICLOBBER**, **FRQSTATE**).
IOUTPUT_MAVG integer	Flag that controls whether or not the model will output monthly-average history files. The fields that are written to these files are a small subset of all possible fields in the model, and are specified and computed in the sourcecode files mem_average_vars.f90 and average_vars.f90. Changing the set of fields can be accomplished

	by straightforward modification of these files. The monthly averages are accumulated each timestep of the model, and are output and then reset at the end of each calendar month of the simulation. If a simulation begins after the beginning of a particular month, the first output file will contain averages only for the portion of that month that was simulated. **IOUTPUT_MAVG** = 0, no monthly-average output files will be generated **IOUTPUT_MAVG** = 1, the model will generate monthly-average output files in HDF5 format. **HFILEPREF** and **ICLOBBER** are used in generating these files, but a model-added character (M) in their filenames distinguishes them from the standard history files.
IOUTPUT_DAVG integer	Flag that controls whether or not the model will output daily-average history files. The fields that are written to these files are a small subset of all possible fields in the model, and are specified and computed in the sourcecode files mem_average_vars.f90 and average_vars.f90. Changing the set of fields can be accomplished by straightforward modification of these files. The daily averages are accumulated each timestep of the model, and are output and then reset at the end of each calendar day of the simulation. If a simulation begins after the beginning of a particular day, the first output file will contain averages only for the portion of that day that was simulated. **IOUTPUT_DAVG** = 0, no daily-average output files will be generated **IOUTPUT_DAVG** = 1, the model will generate daily-average output files in HDF5 format. **HFILEPREF** and **ICLOBBER** are used in generating these files, but a model-added character (D) in their filenames distinguishes them from the standard history files.
IOUTPUT_LITE integer	Flag that controls whether or not the model will output "LITE" history files. LITE files are reduced-size history files whose contents are user-specifiable, and they provide a means to output selected fields at a higher frequency than the main history files in order to reduce disk storage. LITE files can be used, for example, to generate plots of selected fields in a subsequent 'PLOTONLY' run at the higher frequency of the LITE file writes. **IOUTPUT_LITE** = 0, no LITE files will be generated

	IOUTPUT_LITE = 1, the model will generate LITE files in HDF5 format. **HFILEPREF** and **ICLOBBER** are used in generating these files, but a model-added character (L) in their filenames distinguishes them from the standard history files. Additional specification of LITE files is provided by **FRQLITE** and **LITE_VARS**.
IOUTPUT_ LATLON integer	Flag that controls whether or not the model will output "LATLON" history files. LATLON files contain a set of instantaneous and time-averaged model output fields that have been interpolated from the OLAM native hexagonal grid to a latitude-longitude grid, which is useful for further processing and analysis by existing tools such as MATLAB. **IOUTPUT_LATLON** = 0, no LATLON files will be generated **IOUTPUT_LATLON** = 1, the model will generate LATLON files in HDF5 format. **HFILEPREF** and **ICLOBBER** are used in generating these files, but a model-added character (LL) in their filenames distinguishes them from the standard history files. . Additional specification of LATLON files is provided by **FRQLATLON**. The source code file fields2_ll.f90, located in the omodel subdirectory of the OLAM code, performs the selection of fields for interpolation to the latitude-longitude grid. The selection can be changed by modifying this file. In addition, a few parameters are internally set in fields2_ll.f90 to control various parts of the process, and these parameter settings can be changed if desired.
HFILEPREF character	Filename prefix, with path name if applicable, of the history files to be written from the current model run. OLAM appends date and time to this prefix. This same prefix is applied to monthly-average and daily-average history files also, but filenames of each file type are given a unique extra character.
ICLOBBER integer	Flag indicating what the model should do if history files already exist from a prior run, and the model is about to write new files of the same name in the same directory. **ICLOBBER** = 0, The model will stop and not overwrite the older files. **ICLOBBER** = 1,

	The model will continue and write new files over the older files.
ICOMPRESS integer	Indicates level of hdf5 compression to be used in writing history files. Allowable values are 0 (no compression) to 9 (full compression). Compression is useful for reducing the size of history files and conserving storage space, but compression requires additional time. **ICOMPRESS** = 3 is considered a good compromise.
LATLONPLOT integer	Flag that controls whether or not the model will generate plots of fields that have been interpolated to a latitude-longitude grid in preparation for writing LATLON history files. For this option to be active, **IOUTPUT_LATLON** must be set to 1. **LATLONPLOT** = 0, no LATLON fields will be plotted. **LATLONPLOT** = 1, LATLON fields will be plotted. The frequency of these plots corresponds to **FRQLATLON**.
FRQSTATE real	Time interval, in seconds, at which successive history files are to be output from the model. At the start of a new simulation [**RUNTYPE** = 'INITIAL'], the model internal time variable, TIME8, is equal to zero. For a history start [**RUNTYPE** = 'HISTORY'], TIME8 is set to the time value read from the history file specified in **HFILIN**. TIME8 counts up to the value of **TIMMAX** for each run. When TIME8 reaches a multiple of **FRQSTATE**, a model history file is written. A history file is also written out when a model simulation begins and when a model run ends, even if **TIMMAX** is not set to an exact integer multiple of **FRQSTATE**.
FRQLITE real	Time interval, in seconds, at which successive LITE history files are to be output from the model. See additional comments for **FRQSTATE**. **FRQLITE** is used only when **IOUTPUT_LITE** = 1.
FRQLATLON real	Time interval, in seconds, at which successive LATLON files are to be output from the model. See additional comments for **FRQSTATE**. **FRQLATLON** is used only when **IOUTPUT_LATLON** = 1.

HFILIN character	Used only when **RUNTYPE** = 'HISTORY'. **HFILIN** specifies the filename of the history file to be read for initializing the current run. The character string placed in **HFILIN** must be the full name of the history file, including the path name if applicable.
LITE_VARS character array	Specifies which model variables are to be written to the LITE history files. See **IOUTPUT_LITE** and **FRQLITE**.
NADDSC integer	Specifies the number of prognostic scalar fields to be added to the model simulation. The user may want to use scalar fields to represent quantities such as chemical or aerosol pollutants that are not standard prognosed fields in OLAM. The added scalar fields are automatically advected, diffused, and advanced forward in time by setting **NADDSC** to a positive value. The user will need to add routines in the code to specify initial values and to compute any source or sink terms required for the added scalars. The number of scalars that can be added in this way is limited only by computer memory.
ISCAL_MONOT integer	Flag (0 = no, 1 = yes) indicating whether a monotonic flux limiter will be applied to scalar advection. Application of the limiter requires extra computation and often acts to damp prognostic fields, but it also prevents advection from un-physically accentuating local maxima and minima in the fields.
SPLIT_ SCALARS integer	Flag (0=no, 1=yes) indicating whether scalar quantities should be advanced in time in two separate stages, first for non-advective tendencies and second for advective tendencies, rather than advancing once for all tendencies combined. The split advancement can sometimes avoid numerical instabilities when "physics" tendencies such as rapid chemical reactions are involved. It is recommended that splitting NOT be done unless instability of this type is suspected.
VERT_ADV_ORDER integer	Order of accuracy of vertical advection (2=second, 3=third).
HORIZ_ADV_ORDER integer	Order of accuracy of horizontal advection (2=second, 3-third).

CFL_PRTFRQ real	Frequency (interval in seconds) at which the model computes advective Courant number and prints the maximum value found in the domain. A value of zero disables the printing.
DEBUG_FP logical	Flag to detect floating point errors such as overflows, INF's, divide-by-zeros, and Nan's. Use of this diagnostic works best when code is compiled in debug mode. (The compilation choice between optimized or debug mode is selected in the include.mk file.) **DEBUG_FP** = .false., flag turned of **DEBUG_FP =** .true., flag turned on
INIT_NANS logical	Flag to initialize model variables with Nan's in order to catch potential errors. Should be used with **DEBUG_FP** = .true. **INIT_NANS** = .false., flag turned off **INIT_NANS** = .true., flag turned on
RAYF_ZMIN integer	Minimum height above the model's zero reference height (usually sea level), at which the Rayleigh friction absorbing layer for horizontal velocity and potential temperature will be activated. Used with **RAYF_DISTIM** and **RAYF_EXPON**. Rayleigh friction is a Newtonian relaxation toward undisturbed, horizontally homogeneous values. Its purpose is to damp gravity wave and other disturbances that approach the top model boundary, so that they will not be reflected back downward. Rayleigh friction is normally used only when **INITIAL** = 1, which is commonly the case when OLAM is run with a limited area domain (see **MDOMAIN**). Rayleigh friction applied to horizontal velocity and potential temperature is not appropriate for **INITIAL** = 2 or 3 on a global domain. The value assigned to **RAYF_ZMIN** should normally be several km lower than the upper boundary of the model domain (see **HDZ**) and should also be above the region of atmosphere where the most significant dynamic and thermodynamic processes occur (which normally includes all the troposphere and lower stratosphere). Thus, **RAYF_ZMIN** should normally be a height in the middle stratosphere. Rayleigh friction is applied above this level.

RAYF_DISTIM real	Used with **RAYF_ZMIN** and **RAYF_EXPON** to activate Rayleigh friction in the upper region of the model. **RAYF_DISTIM** specifies a dissipation time scale in seconds. **RAYF_DISTIM** represents the e-folding decay time of a disturbance being damped by the Rayleigh friction layer in the absence of any other source or sink terms. This decay rate strictly applies only at the top boundary of the model, and the strength of the damping decreases with height at a rate controlled by **RAYF_EXPON**, reaching zero at height **RAYF_ZMIN**. **RAYF_DISTIM** should be related to the rate at which the model is generating upward propagating gravity waves, and normally ranges from 60 to a few hundred seconds. **RAYF_DISTIM** must be at least twice the value of **DTLONG** for numerical stability.
RAYF_EXPON real	Used with **RAYF_ZMIN** and **RAYF_DISTIM** to specify the shape of the vertical profile of nudging strength. When **RAYF_EXPON** is set to 1, nudging strength increases linearly with height from zero at height **RAYF_ZMIN** to its maximum value at the model top. **RAYF_EXPON** set to 2 causes a quadratic increase, slowly at first and more rapidly later. A value of 1 is most commonly used, but experimentation with values as high as 2 or 3 might demonstrate superior performance.
RAYFW_ZMIN real	Minimum height above the model's zero reference height (usually sea level), at which the Rayleigh friction absorbing layer for vertical velocity will be activated. Used with **RAYFW_DISTIM** and **RAYFW_EXPON**. Rayleigh friction is a Newtonian relaxation toward an undisturbed, horizontally homogeneous value, which for vertical velocity is assumed to be zero. Its purpose is to damp gravity wave and other disturbances that approach the top model boundary, so that they will not be reflected back downward. Rayleigh friction for vertical velocity may be used (and is recommended) with any value of **INITIAL**. The value assigned

	to **RAYFW_ZMIN** should normally be several km lower than the upper boundary of the model domain (see **HDZ**) and should also be above the region of atmosphere where the most significant dynamic and thermodynamic processes occur (which normally includes all the troposphere and lower stratosphere). Thus, **RAYFW_ZMIN** should normally be a height in the middle stratosphere. Rayleigh friction is applied above this level.
RAYFW_DISTIM real	Similar to **RAYF_DISTIM** but applies only to vertical velocity. Used with **RAYFW_ZMIN**.
RAYFW_EXPON real	Similar to **RAYF_EXPON** but applies only to vertical velocity. Used with **RAYFW_ZMIN**.
RAYFDIV_ZMIN real	Minimum height above the model's zero reference height (usually sea level), at which the Rayleigh friction absorbing layer for horizontal divergence will be activated. Used with **RAYFDIV_DISTIM** and **RAYFDIV_EXPON**. Rayleigh friction is a Newtonian relaxation toward an undisturbed, horizontally homogeneous value, which for horizontal divergence is assumed to be zero. Its purpose is to damp gravity wave and other disturbances that approach the top model boundary, so that they will not be reflected back downward. Rayleigh friction for horizontal divergence may be used (and is recommended) with any value of **INITIAL**. The value assigned to **RAYFDIV_ZMIN** should normally be several km lower than the upper boundary of the model domain (see **HDZ**) and should also be above the region of atmosphere where the most significant dynamic and thermodynamic processes occur (which normally includes all the troposphere and lower stratosphere). Thus, **RAYFDIV_ZMIN** should normally be a height in the middle stratosphere. Rayleigh friction is applied above this level.
RAYFDIV_DISTIM real	Similar to **RAYF_DISTIM** but applies only to horizontal divergence. Used with **RAYFDIV_ZMIN**.
RAYFDIV_EXPON real	Similar to **RAYF_EXPON** but applies only to horizontal divergence. Used with **RAYFDIV_ZMIN**.
ISWRTYP integer **ILWRTYP**	Flag specifying options for evaluating shortwave radiative transfer scheme

integer	Flag specifying options for evaluating longwave radiative transfer scheme The options for **ISWRTYP** and **ILWRTYP** are: 0, no radiation 1,2,3 RRTMg parameterization The RRTMg scheme accounts for each form of condensate (cloud water, drizzle, rain, pristine ice, snow, aggregates, graupel, and hail) as well as water vapor, and also uses information on ice crystal habit. (The Harrington radiation scheme is no longer used in OLAM.)
RADFRQ real	Time interval, in seconds, that specifies how often during a model run to update radiative transfer contributions to atmospheric and surface temperature tendencies. The tendencies themselves are applied every timestep of the model integration, but the tendencies are updated only at time intervals specified by **RADFRQ**. Because computing the radiative contribution to temperature tendencies is a relatively expensive computational process, it is suggested to use a value for **RADFRQ** between 600 and 1200. This is sufficient for most situations.
RRTMG_DATADIR character	Specifies the full or relative path of the directory that contains various data files that are read by the RRTMg radiation parameterization. These files are: omic2rrtmg_lw.h5, omic2rrtmg_sw.h5, rrtmg_lw.h5, and rrtmg_sw.h5, and they are normally stored in the /etc subdirectory.
ICFRAC integer	Flag that selects scheme for estimating sub-grid fractional cloudiness associated with grid-scale (microphysics) moisture. The resulting fractional cloudiness value is input to the RRTMg radiation parameterization where it is used in the computation of radiative fluxes. **ICFRAC** currently has no impact on the OLAM solution if the RRTMg scheme is not used. **ICFRAC** = 1 activates the Mocko and Cotton (1995) scheme in which cloud fraction is assumed to increase from 0 to 1 as relative humidity increases from 75% (over ocean) or 85% (over land) to 100%. The increase is nonlinear.

	ICFRAC = 2 activates a scheme introduced by Walko that extends the Mocko and Cotton scheme to grid-scale condensate fields, namely cloud water and pristine ice, in addition to the water vapor field alone. As in the Mocko and Cotton scheme, the moisture field (summed over vapor and condensate) is normalized by saturation vapor density. Walko's scheme assumes a linear increase of cloud fraction with normalized moisture. The minimum and maximum thresholds of normalized moisture that define the range over which cloud fraction increases from 0 to 1 are specified by the user in parameters **CFRACRH1** and **CFRACRH2**. With suitable settings of these parameters, obtained by experimentation, much better radiative fluxes (and other OLAM output fields) can be obtained with **ICFRAC** = 2 than with **ICFRAC** = 1. **ICFRAC** > 2 is similar to **ICFRAC** = 2 except that instead of using **CFRACRH1** and **CFRACRH2** from the namelist, geographically-dependent values of these thresholds are specified in the code. The thresholds are chosen to be a function of latitude and are also dependent on whether the location is over land or ocean. The selection of these thresholds is made in the sourcecode file rrtmg_raddriv.f90 (approximately lines 350 to 420) based on experimentation with a global grid of 200 km horizontal spacing. Options for **ICFRAC** = 3:6 are currently implemented, and others can easily be added.
CFRACRH1 real	Used only when **ICFRAC** = 2 and specifies the normalized moisture value above which cloud fraction begins to increase above 0. Tests suggest that value should be above 0.7 and below 1.0.
CFRACRH2 real	Used only when **ICFRAC** = 2 and specifies the normalized moisture value at which cloud fraction ends its linear increase and reaches 1. Tests suggest that value should be above 1.0 and below 1.7.
CFRACCUP real	Upper bound that is imposed on fractional cloudiness when subgrid-scale (parameterized) convection is present. This value is applied only to grid cells where parameterized convection is active, and in the case of deep convection that is generating convective precipitation, also to subcloud layers. A value of 0.7 for this parameter has been used, but experimentation is encouraged. Used only when RRTMg radiation scheme is activated.

NQPARM integer array	MRL-dependent parameter. Specifies the deep convective parameterization for each MRL. **NQPARM** = 0, no convective parameterization. **NQPARM** = 1, Tiedtke parameterization. **NQPARM** = 2, Grell ensemble parameterization. **NQPARM** = 3, Kain-Fritsch (ETA) parameterization. **NQPARM** = 4, Emanuel parameterization. **NQPARM** = 5, Grell-Freitas parameterization. When convective parameterization is activated, the model will use parameter **CONFRQ**. Convective parameterization is used to vertically redistribute heat, moisture, momentum, and other scalar prognostic variables (see **CONV_UV_MIX** and **CONV_TRACER_MIX**) in a grid column when the model generates a region which is convectively unstable and when the horizontal grid resolution is too coarse for the model to develop its own convective circulation. Ideally, resolving a convective circulation would require at least a few grid cells to horizontally span an updraft, which for deep convection would normally require the horizontal cell size to be less than 1 or 2 kilometers. Coarser resolution than this would make realization of sufficiently strong vertical motion difficult or impossible to adequately bring about the required vertical exchange of heat and moisture so as to convert the convective available potential energy into other forms. Thus, it is on coarser grids where a parameterized convective adjustment becomes necessary. Unfortunately, the convective parameterization schemes currently available generally assume the grid size in the horizontal to be around 20 kilometers or greater. This means that convective parameterization may be activated on any grid of this resolution, but that at resolutions between about 2 and 20 kilometers, no adequate convective adjustment scheme exists.
CONFRQ real	Used when convective parameterization is activated. (see **NQPARM**)

	Time interval, in seconds, during a simulation at which contributions to atmospheric tendencies in temperature and moisture by convective parameterization are to be computed. These tendencies are applied to the temperature and moisture fields every timestep of the model integration, but are updated only at the time interval specified for **CONFRQ**. There are two reasons why the contributions should be updated less frequently than every timestep. One is that a computational expense is involved in the convective parameterization computations, and frequent updates will slow down the model. The second reason has an important physical basis. For example, consider a case of a squall line moving through a grid column. As the squall line enters the grid column, it will begin to stabilize the air that it has moved through. However, the air in front of the squall line is still convectively unstable and will contribute to the maintenance of the squall line. The mean atmosphere, that is, the atmosphere represented on a grid column, soon may be stabilized by the convection. If the convective tendencies were updated in the model every 5 minutes or so, the result for this example is that the original convection might be shut down in its early stages. A more accurate representation of the squall line would be to let it run through the entire grid column. This is achieved by setting **CONFRQ** to a time interval comparable to the amount of time it takes for the squall line to cross the grid column, such as 1200 to 1800 seconds.
CONV_UV_MIX integer **CONV_TRACER_MIX** integer	Flags that specify whether convective parameterization will perform vertical mixing of horizontal momentum (UV) and tracer quantities (TRACER), respectively. 0, no 1, yes
IDIFFK integer array	MRL-dependent parameter. Flag to control the type of parameterization used for computing both horizontal and vertical diffusion coefficients. **IDIFFK** = 0, no turbulent diffusion **IDIFFK** = 1,

	Vertical diffusion coefficients in the PBL are based on the ACM2 scheme and on the Smagorinsky scheme (see IDIFFK = 2) elsewhere. **IDIFFK** = 2, Vertical and horizontal turbulent diffusion coefficients are based on the Smagorinsky scheme in which strain rate is evaluated from vertical gradients of horizontal wind, and vertical length scale evaluated from the local vertical grid spacing times namelist parameter **CSZ**, the horizontal grid spacing times **CSX,** height above ground, and an asymptotic limiting value. In addition, modifications of the vertical diffusion coefficient due to static stability or instability are used, based on formulations of Hill and Lilly. The Lilly modification is in the form of a multiplying factor, equal to sqrt (1-R kh/km) where R is the Richardson number and kh/km is the ratio of the scalar to momentum vertical diffusion coefficients specified by the user in namelist parameter **ZKHKM**. The multiplying factor is greater than 1 in unstable cases (i.e., where wind shear and/or unstable lapse rates make the Richardson number sufficiently low, and is less than 1 in stable case. The Hill modification applies only to regions of unstable lapse rates (having negative squared Brunt-Vaisala frequencies), and consists of adding the absolute value of the Brunt-Vaisala frequency squared to the deformation rate, to obtain a modified inverse time scale for the diffusion coefficient computation. The Lilly and Hill modifications were originally designed for use without each other, although the added vertical diffusion in unstable air obtained by using both together is usually desirable. **IDIFFK** = 3, Same as for **IDIFFK** = 2 except that a 3D rate-of-strain scalar value is used instead of the vertical gradient of the horizontal wind. This option should be used for Large Eddy Simulation (LES), but is not necessary when horizontal grid spacing is significantly larger than vertical spacing.
CSX real array	MRL-dependent parameter. Used when **IDIFFK** = 2. **CSX** is a coefficient used in computing the horizontal diffusive length scale, which is defined as the product of **CSX** and the horizontal grid spacing.
CSZ	MRL-dependent parameter. Used when **IDIFFK** = 2.

real array	**CSZ** is a coefficient used in computing the vertical diffusive length scale, which is defined as the product of **CSZ** and the vertical grid spacing.
AKMIN real array	MRL-dependent parameter. Used when **IDIFFK** = 2 or 3. **AKMIN** is used to impose a minimum value for the horizontal diffusion coefficients throughout the model domain. **AKMIN** = 0 results in a minimum diffusion coefficient of zero. Originally designed for the RAMS model, minimum horizontal diffusion is proportional to **AKMIN** and is scaled such that if **AKMIN** = 1.0, minimum horizontal diffusion coefficients were approximately what was required for computational stability if no other damping method is used. However, OLAM provides damping in the advection algorithm and it is normally recommended to set **AKMIN** = 0.
MICLEVEL integer	Flag that specifies the level of moisture complexity to be activated in the model. This flag is the primary means by which the user tells the model whether to consider the effects of moisture in the simulation, and if so, to what degree. **MICLEVEL** = 0, Causes the model to run dry, completely eliminating any process which influences or is influenced by any phase of moisture. When **MICLEVEL** = 0, **ISWRTYP** and **ILWRTYP** must be set to 0. **MICLEVEL** = 1, Water vapor is included in the model, but no form of condensate. The effects of vapor on air density and radiative transfer are represented. **MICLEVEL** = 2, Activates condensation of water vapor to cloud water wherever supersaturation is attained. The partitioning of the total water substance into vapor and cloud water is purely diagnostic in this case. No other forms of liquid or ice water are considered. Both the density effect of water vapor and the liquid water loading of cloud water are included in the vertical equation of motion. Radiative effects of both water vapor and cloud water are activated if the radiation parameterization is itself activated. **MICLEVEL** = 3,

	Activates the bulk microphysics parameterization, which includes cloud water, drizzle, rain, pristine ice, snow, aggregates, graupel, and hail, or certain subsets of these as specified by **ICLOUD**, **IDRIZ**, **IRAIN**, **IPRIS**, **ISNOW**, **IAGGR**, **IGRAUP**, and **IHAIL**. This parameterization includes the precipitation process.
ICLOUD integer **IDRIZ** integer **IRAIN** integer **IPRIS** integer **ISNOW** integer **IAGGR** integer **IGRAUP** integer **IHAIL** integer	Microphysics flags. Used only if **MICLEVEL** = 3. These eight parameters control the activation of condensate categories. Activation means that the mixing ratio of the particular condensate category is prognosed from conservation equations which include advective, diffusive, and precipitation tendencies, and source terms resulting from interactions between the various condensate categories. Independent selection of the eight condensate categories to be activated versus not activated is described below. Not all possible combinations are intended as valid nor would they function properly without code modification. Activation of all categories simultaneously is the normal practice, although activation of the limited sets {cloud}, {cloud, drizzle, rain}, {cloud, pristine ice}, {pristine ice}, {pristine ice, snow}, {pristine ice, aggregates}, and {pristine ice, snow, aggregates} are reasonable when only a limited range of microphysical processes need to be considered. Parameter — Flag Options — Condensate category **ICLOUD** — 0, 4, 5 — cloud droplets **IDRIZ** — 0, 5 — drizzle **IRAIN** — 0, 2, 5 — rain **IPRIS** — 0, 5 — pristine ice **ISNOW** — 0, 2, 5 — snow **IAGGR** — 0, 2, 5 — aggregates **IGRAUP** — 0, 2, 5 — graupel **IHAIL** — 0, 2, 5 — hail Flag 0 has the same meaning for all seven condensate categories: The condensate category is not activated, and any process involving the interaction of that category with other condensate categories is not performed.

ICLOUD

Flag 4

The number concentration of cloud droplets, when cloud water is present, is set equal to the concentration of cloud condensation nuclei (CCN). This is true whether CCN are themselves of specified concentration or are prognosed (see **ICCN** and **CCNPARM**). Cloud droplet mean diameter is diagnosed from this concentration and the prognosed mixing ratio.

Flag 5

The number concentration of cloud droplets is prognosed. The number of nucleated cloud droplets is based on the concentration and physical properties of cloud condensation nuclei (CCN), as well as on supersaturation production rate and other environmental factors. CCN must be prognosed for this value of **ICLOUD** (see **ICCN** and **CCNPARM**). Cloud droplet mean diameter is diagnosed from the prognosed concentration and the prognosed mixing ratio.

IDRIZ

Flag 5

The number concentration of drizzle droplets is prognosed. The number of nucleated drizzle droplets is based on the concentration of giant cloud condensation nuclei (GCCN). This is true whether GCCN are themselves of specified concentration or are prognosed (see **IGCCN** and **GCCNPARM**). Drizzle droplet mean diameter is diagnosed from the prognosed concentration and the prognosed mixing ratio.

IPRIS

Flag 5

The number concentration of pristine ice is prognosed. The number of nucleated pristine ice crystals is based on the concentration of ice forming nuclei (IFN) as well as on the supersaturation and temperature. This is true whether IFN are themselves of specified concentration or are prognosed (see **IIFN** and **IFNPARM**). Pristine ice mean diameter is diagnosed from the prognosed concentration and the prognosed mixing ratio.

	IRAIN, ISNOW, IAGGR, IGRAUP, IHAIL Flag 2 The mean diameter is specified is specified in the associated namelist parameter (**RPARM, SPARM, APARM, GPARM, HPARM)**, and the number concentration is diagnosed automatically from this mean diameter and the prognosed mixing ratio. Flag 5 Number concentration is prognosed. Mean diameter is diagnosed from the prognosed mixing ratio and number concentration. Note: If any of the parameters **IRAIN, ISNOW, IAGGR, IGRAUP**, or **IHAIL** are set to 5, the model will override the settings of these parameters and set them all to 5. However, any that are set to 0 will remain 0. That is, the model requires that if number concentration is predicted for any of these categories- rain, snow, aggregate, graupel, or hail - it must be predicted for all of these categories that are active.
ICCN integer	If **ICCN**=1, cloud condensation nuclei (CCN) are not prognosed but are of specified concentration that is constant in time but (may be) variable in space. The concentration is determined in part from **CCNPARM** and in part from a vertical profile that is specified in subroutine init_nuc_factors, which is in the source code file micro_coms.f90, and can be modified by editing the code. If **ICCN** >= 2, CCN are prognosed. The number of individually prognosed CCN species and their physical characteristics are specified in subroutine ccnbin_init, which is in the source code file omic_nuc.f90, according to the chosen value of **ICCN**. OLAM currently contains source functions for dust and sea salt types of CCN, which provides at least one means of re-generating these CCN populations as CCN are scavenged by various mechanisms. Addition of other CCN types generally requires source functions as well.

IGGCN integer	If **IGCCN** = 1, giant cloud condensation nuclei (GCCN) are not prognosed but are of specified concentration that is constant in time but (may be) variable in space. The concentration is determined in part from **GCCNPARM** and in part from a vertical profile that is specified in subroutine init_nuc_factors, which is in the source code file micro_coms.f90, and can be modified by editing the code. If **IGCCN** = 2, GCCN are prognosed. OLAM currently contains a sea salt source function for GCCN.
IIFN integer	If **IIFN** = 1, ice forming nuclei (IFN) are not prognosed but are of specified concentration that is constant in time but (may be) variable in space. The concentration is determined in part from **IFNPARM** and in part from a vertical profile that is specified in subroutine init_nuc_factors, which is in the source code file micro_coms.f90, and can be modified by editing the code. If **IIFN** = 2, IFN are prognosed. OLAM currently contains no source function for IFN, and such a source should be added if one wants to prognose IFN and provide a means for maintaining the IFN population as IFN are scavenged by various mechanisms. However, large dust, which is prognosed as a CCN type and has an existing source function, also functions as IFN, so pristine ice crystals can still form without this separate IFN category (controlled by **IIFN**).
RPARM real	Rain droplet mean diameter, in meters. Used when **IRAIN** equals 2.
SPARM real	Snow hydrometeor mean diameter, in meters. Used when **ISNOW** equals 2.
APARM real	Aggregate mean diameter, in meters. Used when **IAGGR** equals 2.
GPARM real	Graupel mean diameter, in meters. Used when **IGRAUP** equals 2.
HPARM real	Hail mean diameter, in meters. Used when **IHAIL** equals 2.

CCNPARM real	Concentration (#/kg_air) of cloud concentration nuclei (CCN) in the case where **ICCN** is set to 1. Specifically, the concentration specified in **CCNPARM** applies at sea level, and the concentration by default decreases with height as described under **ICCN**. If **CCNPARM** is set to zero, the sea level concentration is obtained from a geographically-based dataset of mean observed cloud droplet concentrations. **CCNPARM** is not used when **ICCN** is set to 2 or greater.
GCCNPARM real	Concentration (#/kg_air) of giant cloud concentration nuclei (GCCN) in the case where **IGCCN** is set to 1. Specifically, the concentration specified in **GCCNPARM** applies at sea level, and the concentration by default decreases with height as described under **IGCCN. GCCNPARM** is not used when **IGCCN** is set to 2.
IFNPARM integer	Concentration (#/kg_air) of ice forming nuclei (IFN) in the case where **IIFN** is set to 1. Specifically, the concentration specified in **IFNPARM** applies at sea level, and the concentration by default decreases with height as described under **IIFN. IFNPARM** is not used when **IIFN** is set to 2.
CO2FLAG integer	Flag indicating whether CO2 will be included as a prognostic variable (0 = no; 1 = yes). OLAM version 5.3 is the first to provide this option in OLAMIN. In future versions, prognosed CO2 will be made to interact with the radiative transfer parameterization and will include source and sink terms. At present, the implementation excludes these aspects. All that is done is initialization (at a default concentration of 400 ppmv relative to dry air) and application of advective and diffusive transport.
NSNDG integer	Number of sounding levels to be read into arrays **PS**, **TS**, **RTS**, **US**, and **VS**. Used only when the parameter **INITIAL** is set to 1. When **INITIAL** is set to 1, the model atmospheric fields are initialized horizontally homogeneously from a single sounding. The parameters **NSNDG**, **IPSFLG**, **ITSFLG**, **IRTSFLG**, **IUSFLG**, **HS**, and **SOUNDING** (which gets copied to **PS**, **TS**, **RTS**, **US**, and **VS** arrays in the model code) specify the input sounding values that are used to initialize a simulation when the parameter **INITIAL** is set to 1.

IPSFLG integer	Flag that specifies how to interpret **PS** values. The **PS** value defines the height or pressure of the sounding level. **IPSFLG** = 0, **PS** is pressure in millibars **IPSFLG** = 1, **PS** is height in meters.
ITSFLG integer	Flag that specifies how to interpret **TS** values. **ITSFLG** = 0, **TS** is temperature in degrees C, **ITSFLG** = 1, **TS** is temperature in degrees K, **ITSFLG** = 2, **TS** is potential temperature in degrees K.
IRTSFLG integer	Flag that specifies how to interpret **RTS** values. **IRTSFLG** = 0, **RTS** is dew point temperature in degrees C, **IRTSFLG** = 1, **RTS** is dew point temperature in degrees K, **IRTSFLG** = 2, **RTS** is water vapor mixing ratio in grams per kilogram of air, **IRTSFLG** = 3, **RTS** is relative humidity in percentage, **IRTSFLG** = 4, **RTS** is dew point depression in degrees K.
IUSFLG integer	Flag that specifies how to interpret **US** and **VS** values. **IUSFLG** = 0, **US** are components of velocity in meters per second in the x-direction, and **VS** are components of velocity in meters per second in the y-direction. **IUSFLG** = 1, **US** are wind directions in degrees azimuth clockwise from true north, and **VS** are values are interpreted as wind speeds in meters per second.
HS real	Absolute height, in meters, relative to sea level, of the first (lowest) sounding level. **HS** must be at or below the lowest topography height in the model domain. This may require adding one or more levels to the bottom of a real sounding.
P_SFC real	Pressure, in mb, of the lowest sounding level. Used only when **ISPFLG** = 1.
SOUNDING real array	A two-dimensional array that contains sounding level data read by OLAM to initialize model atmospheric fields when **INITIAL** = 1.

	Each row of values listed for **SOUNDING** must contain 5 values representing the quantities **PS**, **TS**, **RTS**, **US**, and **VS,** in the order listed. The values within each line must be separated by commas. The top line in the listing represents the lowest sounding level; the second line represents the next higher sounding level, and so on. Inside the model, the values read from **SOUNDING** are copied to the individual arrays **PS**, **TS**, **RTS**, **US**, and **VS**. Use of **SOUNDING** as an array facilitates listing these individual arrays in column form in OLAMIN while still reading their values in namelist format. The total number of sounding levels to be read and used is specified in **NSNDG**.
PS real array	Sounding level height or pressure. (see **ISPFLG** and **SOUNDING**) The values given for **PS** will determine the vertical spatial resolution of the sounding, and should be chosen to properly represent any significant levels. The sounding data should be at least as low as the lowest topography in the domain, and extend to a height greater than the top of the model domain. Therefore, the top sounding level must contain either a height or a pressure higher in the atmosphere than the model domain top. The model domain top is either specified by parameter **ZZ**, or determined by the combination of parameters **HDZ** and **DZ**.
TS real array	Sounding level temperature value. (see **ITSFLG** and **SOUNDING**) The user should use caution in specifying sounding temperatures to prevent inadvertent superadiabatic layers. High vertical resolution is sometimes desirable in a sounding because the model fields of temperature and moisture mixing ratio are interpolated linearly from it in the vertical direction, and relative humidity is a very nonlinear function of these two interpolated fields. This can result in supersaturated layers in the model initial fields where none exist in the sounding.
RTS real array	Sounding level moisture value. (see **IRTSFLG** and **SOUNDING**)
US	Sounding level wind value. (see **IUSFLG** and **SOUNDING**)

real array	
VS real array	Sounding level wind value. (see **IUSFLG** and **SOUNDING**)
GRIDFILE character **SFCGFILE** character	Filename prefixes, with path names if applicable, for the grid file and surface grid file generated at the beginning of a model simulation. **GRIDFILE** contains the number, coordinates, size, and other geometric parameters of the atmospheric grid, including local connectivity between adjacent grid cells. **SFCGFILE** contains the number, coordinates, size, interconnectedness, and connectivity to the atmospheric grid of all surface grid cells, combining those representing land, lakes, and oceans. It also specifies whether each cell in the grid is land, lake, or sea, and contains additional time-independent information for each of these types. Both of these files are generated in a model run with **RUNTYPE** = 'MAKEGRID' and are read in subsequent runs with **RUNTYPE** = 'INITIAL', 'HISTORY', or 'PLOTONLY'.
ISFCL integer	Flag that specifies whether or not to activate LEAF4. LEAF4 is the OLAM submodel for energy and moisture budgets in soil, snowcover, vegetation, and canopy air, and for surface fluxes from these and from water surface areas. **ISFCL** = 0, LEAF4 submodel is not activated. Surface drag coefficients and surface fluxes of heat and vapor are assigned default values of zero in subroutine surface_turb_flux in *surface_fluxes.f90*. These defaults may be overridden by modifying the subroutine. **ISFCL** = 1, Activate the LEAF4 submodel. LEAF4 prognoses soil, vegetation, snowcover, and canopy air temperature and moisture based on vertical diffusion and exchange with the atmosphere. (see **NZG**, **NVGCON**, **NSLCON**, **SLZ**, and **RUNTYPE** = 'MAKEGRID')

NZS integer	Specifies the maximum number of snowcover layers allowed to occur in LEAF4. **NZS** must be at least 1. If no snowcover is expected in a simulation, there is no need for **NZS** to be greater than 1. If snowcover will occur, and if its evolution is important to the simulation, a **NZS** value around 5 is suggested. This allows multiple snow layers when snowcover is sufficiently deep.
NZG integer	Specifies the number of soil layers to be used in OLAM-SOIL. OLAM-SOIL is a submodel of OLAM that prognoses energy and moisture in soil, snowcover, vegetation, and canopy air. The soil model component consists of a grid representing finite volumes in the ground to a depth of tens or hundreds of meters. Soil temperature and moisture are prognosed on this grid based on equations governing their internal transport and external exchange with the atmosphere. OLAM-SOIL is activated by the parameter **ISFCL.** The thickness of each soil grid level is determined by the three parameters **NZG**, **LANDGRID_DZTOP**, and **LANDGRID_DEPTH**, and by the mathematical rule that grid level thicknesses increase geometrically with increasing depth in the ground. A constant stretch factor gives the thickness of each layer relative to the one above it. **NZG** can be set to about 25 for **LANDGRID_DEPTH** = 100 (m) and **LANDGRID_DZTOP** = 0 .05 (m). Larger **LANDGRID_DEPTH** and/or smaller **LANDGRID_DZTOP** should use larger **NZG**. For any case, larger **NZG** decreases the geometric stretch ratio between consecutive soil grid layers. The soil grid is set up only when **RUNTYPE** = 'MAKEGRID'. In order to change the soil grid geometry, it is necessary to perform a new 'MAKEGRID' simulation.
LANDGRID_DZTOP real	Thickness, in meters, of the top (shallowest) soil grid level. Thicknesses of all other layers are determined by the combination of **NZG**, **LANDGRID_DZTOP**, and **LANDGRID_DEPTH**. It is suggested to set **LANDGRID_DZTOP** to about 0.05 (m), which is thin enough to allow soil moisture and temperature to respond within several minutes to most changes at the surface, and thick enough to avoid numerical instabilities associated with low

	heat storage capacitance.
LANDGRID_DEPTH real	Depth, in meters, of the bottom of the soil model grid. Depths and thicknesses of each soil grid level are determined by the combination of **NZG**, **LANDGRID_DZTOP**, and **LANDGRID_DEPTH**. It is suggested to set **LANDGRID_DEPTH** to about 100 m, although some applications may require much deeper representation of the soil grid while for others a soil only a few m deep may be appropriate.
ISOILFLG integer	Flag that specifies how to initialize soil textural class and/or soil composition. **ISOILFLG** = 1, Soil composition is read from SoilGrids data files, and deeper ground permeability and porosity are read from GLHYMPS database. Soil hydraulic properties are determined from this compositional data based on the specification in **ISOILPTF**. The prefixes for all filenames in these databases are specified in **SOILGRIDS_DATABASE** and **GLHYMPS_DATABASE**, respectively. **ISOILFLG =** 2, Soil textural class is based on USDA classification, which is determined from FAO soil database. The USDA classification provides both soil composition values and van Genuchten soil hydraulic properties; how these are used depends on the specification of **ISOILPTF**. The prefix for all filenames in the FAO database is specified in **SOIL_DATABASE**.
ISOILPTF integer	Flag that specifies how to determine soil (van Genuchten) hydraulic properties. **ISOILPTF** = 1, Use Vereecken pedotransfer functions and GLHYMPS permeability and porosity at deeper levels. **ISOILPTF** = 2, Use de Boer pedotransfer functions and GLHYMPS permeability and porosity at deeper levels. **ISOILPTF** = 3, Use USDA specification of van Genuchten parameters **ISOILPTF** = 3 may only be used when **ISOILFLG** = 2.

ITOPOFLG integer	Flag that specifies how to initialize topography. **ITOPOFLG** = 1, Topography is interpolated to the grid from a topography database, A prefix for all filenames in this database is specified in **TOPO_DATABASE**. **ITOPOFLG** = 2, Used if the simulation requires idealized or flat topography. OLAM calls subroutine topo_init in the model file *mksfc_topo.f90* to assign the terrain height, which by default sets terrain height to zero for the entire domain. The user can override this default or customize terrain height by editing the subroutine.
IVEGFLG integer	Flag that specifies how to initialize vegetation type. **IVEGFLG** = 1, Vegetation data are obtained from a vegetation database. The prefix for all filenames in this database is specified in **VEG_DATABASE**. **IVEGFLG** = 2, OLAM calls subroutine makesfc in model file *makesfc.f90* that by default sets vegetation type for the entire domain to the value specified in namelist parameter **NVGCON**. The user can override this default or customize vegetation type by editing the subroutine.
NDVIFLG integer	Flag that specifies how to initialize the NDVI (Normalized Difference Vegetation Index). **NDVIFLG** = 1, NDVI is read from a database. The prefix for all filenames in this database is specified in **NDVI_DATABASE.** **NDVIFLG =** 2, OLAM calls subroutine makesfc in model file *makesfc.f90* that by default sets NDVI to .5 for the entire domain. The user can override this default or customize NDVI by editing the subroutine.
ISSTFLG integer	Flag that specifies how to initialize sea surface temperature (SST).

	SST is important for determining the fluxes of heat and moisture between the surface and atmosphere. Note: SST is a variable defined for each surface grid cell in the model and is relevant for any surface grid cell which is not completely covered by land. OLAM currently uses SST to represent the surface temperature of any permanent body of water in the grid cell, whether it is an ocean, lake, or river. **ISSTFLG** = 0, OLAM calls subroutine makesfc in model file *makesfc.f90* that by default sets the SST for the entire domain to the value specified in namelist parameter **SEATMP**. The user can override this default or customize SST by editing the subroutine. **ISSTFLG** = 1, SST is interpolated from a SST database. The prefix for all filenames in this database is specified in **SST_DATABASE**. **ISSTFLG** = 2, SST is interpolated from degribbed analysis files (e.g., a CFSR dataset obtained from NOAA). Requires that **INITIAL** = 3 and that SST be present on the analysis file.
ISEAICEFLG integer	Flag that specifies how to initialize SEAICE (the sea ice fractional coverage). SEAICE influences the radiative fluxes at the sea surface, but does not currently have any other effect in the model. **ISEAICEFLG** = 0, OLAM calls subroutine makesfc in model file *makesfc.f90* that by default sets the SEAICE values for the entire domain to zero. The user can override this default or customize SEAICE by editing the subroutine. **ISEAICEFLG** = 1, SEAICE ice is interpolated from a sea ice database. The prefix for all filenames in this database is specified in **SEAICE_DATABASE**. **ISEAICEFLG** = 2, SEAICE ice is interpolated from degribbed analysis files (e.g., a CFSR dataset obtained from NOAA). Requires that **INITIAL** = 3 and that SEAICE be present on the analysis file.
ISOILSTATEINIT	Flag that specifies how to initialize soil moisture.

integer	**ISOILSTATEINIT** = 0, Soil moisture is initialized from **SLMSTR**. **ISOILSTATEINIT** = 1, Soil moisture is read in from file, whose name and path are defined in **SOILSTATE_DB**. **ISOILSTATEINIT** = 2, Soil moisture is read from initial analysis file files (e.g., a CFSR dataset obtained from NOAA). Requires that **INITIAL** = 3 and that SEAICE be present on the analysis file.
IWATERTABFLG integer	Flag that specifies how to obtain water table depth, which influences vertical flux of water through the bottom boundary of the soil model. A shallow water table, for example, can provide a supply of moisture from below even during prolonged atmospheric drought conditions. **IWATERTABFLG** = 0, set water table depth according to default assigned to landuse class for each location. **IWATERTABFLG** = 1, read water table depth from file whose name and path are specified in **WATERTAB_DB**.
IOROGSLOPEFLG	Flag that specifies whether to read orographic roughness information from a file. (0 = no; 1 = yes).
TOPO_DATABASE character	Specifies the filename prefix, with pathname if applicable, of the topography database. Used when **ITOPOFLG** = 1. The standard OLAM topography database contains data at 30 arc-second intervals of latitude and longitude for all areas of the globe. Using this standard database is by far the most common way to initialize topography in OLAM. The data files are named according to a convention that identifies the latitude and longitude of their southwest corners. For example, files in the topography database have names such as TM30N120W, indicating a southwest corner at 30 degrees north latitude and 120 degrees west longitude, with the characters 'TM' serving as a filename prefix for all files in the dataset. The value specified for **TOPO_DATABASE** would be 'TM' in this case, as the model supplies the remainder of the file name according to the file(s) it needs to read.

VEG_DATABASE character	Specifies the filename prefix, with pathname if applicable, of the vegetation database. Used when **IVEGFLG** = 1. The standard OLAM vegetation database is currently available at 30 arc-second intervals of latitude and longitude for all areas of the globe. **VEG_DATABASE** and the following parameters for other DATABASE types are named according to a convention that identifies the latitude and longitude of their southwest corners. For example, one file in the vegetation database has the name OGE2_30N120W, indicating a southwest corner at 30 degrees north latitude and 120 degrees west longitude. The characters 'OGE2_' serve as a filename prefix for this and all other files in the dataset. The value specified for **VEG_DATABASE** would be 'OGE2_' in this case, as the model supplies the remainder of the file name according to the file(s) it needs to read.
SOIL_DATABASE character	Specifies the filename prefix, with pathname if applicable, of the soil textural class database. Used when **ISOILFLG** = 2. The standard OLAM soil database is currently available at 2 arc-minute intervals of latitude and longitude for all areas of the globe.
SOILGRIDS_DATABASE character	Specifies the filename prefix, with pathname if applicable, of the SoilGrids soil composition database. Used when **ISOILFLG** = 1.
GLHYMPS_DATABASE character	Specifies the filename prefix, with pathname if applicable, of the GLHYMPS ground permeability and porosity database. Used when **ISOILFLG** = 1.
NDVI_DATABASE character	Specifies the filename prefix, with pathname if applicable, of the NDVI database. Used when **NDVIFLG** = 1. The standard OLAM NDVI database is currently available at 30 arc-second intervals of latitude and longitude for all areas of the globe and for every month of the year.
WATERTAB_DB character	Specifies the name and path of the file containing water table depth information. Used when **IWATERTABFLG** = 1.

OROG_SLOPE_DB character	Specifies the name and path of the file containing orographic roughness information. Used when **IOROGSLOPEFLG** = 1.
SST_DATABASE character	Specifies the filename prefix, with pathname if applicable, of the sea surface temperature (SST) database. Used when **ISSTFLG** = 1. The standard OLAM SST database contains data at 1 degree intervals of latitude and longitude for all areas of the globe and for every month of the year.
SEAICE_ DATABASE character	Specifies the filename prefix, with pathname if applicable, of the sea ice (SEAICE) database. Used when **ISEAICE** = 1.
IUPDNDVI integer	Flag to indicate whether to hold observed NDVI values at their initial values in the simulation or to update the NDVI values from time-dependent observations. **IUPDNDVI** = 0, Observed NDVI values are held constant in time for a model run. **IUPDNDVI** = 1, Observed NDVI values are continually linearly interpolated in time between the nearest observing times before and after the current model runtime.
IUPDSST integer	Flag to indicate whether to hold observed SST values at their initial values in the simulation or to update the SST values from time-dependent observations. **IUPDSST** = 0, Observed SST values are held constant in time for a model run. **IUPDSST** = 1, Observed SST values are continually linearly interpolated in time between the nearest observing times before and after the current model runtime.
IUPDSEAICE integer	Flag to indicate whether to hold observed SEA ICE values at their initial values in the simulation or to update the SEA ICE values from time-dependent observations. **IUPDSEAICE** = 0,

	Observed SEA ICE values are held constant in time for a model run. **IUPDSEAICE** = 1, Observed SEA ICE values are continually linearly interpolated in time between the nearest observing times before and after the current model runtime.
SEATMP real	Sea surface temperature, in Kelvins. **SEATMP** is the default value used when **ISSTFLG** = 0. This default value is assigned to all grid cells in subroutine makesfc in makesfc.f90. The user can override this default or customize sea surface temperature by editing the subroutine. When **SEATMP** is used, OLAM assumes that the sea surface temperature is constant in time during a simulation. Note: SST is a variable defined for each surface grid cell in the model and is relevant for any surface grid cell which is not completely covered by land. OLAM currently uses SST to represent the surface temperature of any permanent body of water in the grid cell, whether it is an ocean, lake, or river.
SEAICE real	Sea ice fractional coverage (range from 0 to 1) if applied horizontally homogeneously to all grid cells. Used only when **ISEAICEFLG** = 0. The user can override this default or customize sea ice fractional coverage in subroutine makesfc in makesfc.f90. When **SEAICE** is used, OLAM assumes that it remains constant in time.
NVGCON integer	Specifies the landuse class used in LEAF3. **NVGCON** is the default value used when **IVEGFLG** = 2. This default value is assigned to all grid cells in subroutine makesfc in model file makesfc.f90. The user can override this default or customize landuse class by editing the subroutine. 0 Ocean 1 Lakes, rivers, streams 2 Ice cap/glacier 3 Desert, bare soil 4 Evergreen needleleaf tree 5 Deciduous needleleaf tree 6 Deciduous broadleaf tree 10 Semi-desert 11 Tundra 12 Evergreen shrub 13 Deciduous shrub 14 Mixed woodland 15 Crop/mixed farming, C3 grassland 16 Irrigated crop

	7 Evergreen broadleaf tree 8 Short grass 9 Tall grass 17 Bog or marsh 18 Wooded grassland 19 Urban and built up 20 Wetland evergreen broadleaf tree Each of these landuse classes is assigned a set of land surface parameter (LSP) values, including leaf area index, vegetation fractional coverage, vegetation height, albedo, and root depth. The LSP values are defined in a data statement in subroutine sfcdata in the file leaf3_startup.f90.
NSLCON integer	Species the soil textural class used in LEAF3. NSLCON is the default value used when ISOILFLG = 2. This default value is assigned to all grid cells in subroutine makesfc in model file makesfc.f90. The user can override this default or customize soil textural class by editing the subroutine. The soil type controls soil properties that ultimately affect the influence by the soil on the atmosphere. These properties are thermal diffusivity, specific heat, moisture capacity (porosity), and moisture diffusivity. The thermal properties are highly influenced by the moisture content of the soil, but the particular relation between the thermal properties is dependent on the soil class 1 Sand 2 Loamy sand 3 Sandy loam 4 Silt loam 5 Loam 6 Sandy clay loam 7 Silty clay loam 8 Slay loam 9 Sandy clay 10 Silty clay 11 Clay 12 Peat
ED2_ACTIVE integer	Flag (0 = no, 1 = yes) indicating whether ED2 dynamic vegetation model will be used as part of the OLAM simulation. If ED2 is used, parameters in a separate namelist file (defined in **ED2_NAMELIST** and usually called ED2IN) must also be set; these are documented separately.
ED2_NAMELIST character	Indicates namelist filename for case where **ED_ACTIVE** = 1.
IAPR character	Filename, with path name if applicable, that lists input pressure-level analysis files that are to be read by OLAM. All files listed will be listed in an internal model table, and the file will be read if

	and when the model simulation reaches the point in time when the file applies.
DO_CHEM **LTNG_NOX** **PHOTFRQ** **CORE_SHELL_PHOTOLYSIS** **MIE_CALC_PHOTOLYSIS** **EMIS_DIR** **GEIA_EMIS_FILE** **O3NUDFLAG** **O3NUDPRESS** **O3TNUDCENT**	These parameters control operation of the CMAQ chemistry model which is not yet fully operational in OLAM.
NPLT_FILES integer	Number of history files to read and plot fields from or to re-combine from parallel to sequential form. The actual history files to be read are listed in PLT_FILES. Used only when RUNTYPE = 'PLOTONLY'. If monthly-average or daily-average history files are to be plotted in a 'PLOTONLY' run (see NMAVG_FILES and NDAVG_FILES), doing so will override (prevent) any plotting of the standard history files specified in NPLT_FILES and PLT_FILES.
PLT_FILES character array	A list of filenames, with path names if applicable, of previously generated history files to read in for plotting or for combination into single (sequential) files. Used only when **RUNTYPE** = 'PLOTONLY'. If multiple history files are to be read, they must be listed on consecutive lines of OLAMIN, one file per line. Each file in the list is a character string consisting of the file name with directory path if applicable. If the run that produced the history files was in parallel using MPI and WITHOUT using parallel I/O (see **IPAR_OUT**), multiple parallel files were written at each output time, and each parallel file is given a unique filename extension. This extension is NOT to be included in the filename specification for **RUNTYPE**. Only one file listing is used for each simulation

	time, where the character string to be listed is the set of characters that are common to all parallel files for that time. The total number of history files to be read and used for plotting is specified in **NPLT_FILES**.
NMAVG_FILES **integer**	Number of monthly-average history files to read and plot fields from. The actual monthly-average history files to be read are listed in **MAVG_FILES.** Used only when **RUNTYPE** = 'PLOTONLY'.
MAVG_FILES character array	A list of filenames, with path names if applicable, of previously generated monthly-average history files to read in and plot. Used only when **RUNTYPE** = 'PLOTONLY' and **NMAVG_FILES** > 0. If multiple monthly-average history files are to be read, they must be listed on consecutive lines of OLAMIN, one file per line. Each file in the list is a character string consisting of the file name with directory path if applicable. The total number of monthly-average history files to be read and used for plotting is specified in **NMAVG_FILES**.
NDAVG_FILES integer	Number of daily-average history files to read and plot fields from. The actual daily-average history files to be read are listed in **DAVG_FILES.** Used only when **RUNTYPE** = 'PLOTONLY'. If monthly-average history files are to be plotted in a 'PLOTONLY' run (see **NMAVG_FILES**), doing so will override (prevent) any plotting of the daily history files specified in **NDAVG_FILES** and **DAVG_FILES**.
DAVG_FILES character array	A list of filenames, with path names if applicable, of previously generated daily-average history files to read in and plot. Used only when **RUNTYPE** = 'PLOTONLY' and **NDAVG_FILES** > 0. If multiple daily-average history files are to be read, they must be listed on consecutive lines of OLAMIN, one file per line. Each

	file in the list is a character string consisting of the file name with directory path if applicable. The total number of daily-average history files to be read and used for plotting is specified in **NDAVG_FILES**.
NPLT integer	Total number of fields to plot at each plot time. Used when **RUNTYPE** = 'INITIAL, 'HISTORY', or 'PLOTONLY'. The instructions for plotting each field are specified by the parameters following **FLDNAME**. Plots are generated by NCAR graphics and are written to an output file in either metacode, postscript, or pdf format (see **PLTTYPE**). The output filename is specified in **PLTNAME.**
FRQPLT integer	Time interval in seconds between successive plot times. Used when **RUNTYPE** = 'INITIAL' or 'HISTORY'.
DTVEC real	Time interval, in seconds. **DTVEC** controls the length of the plotted vector. Length of the plotted vector **= DTVEC** * local wind speed, where this length refers to a geographic distance before it is mapped to the plot frame. For example, if the wind speed is 20 m/s and DTVEC is set to 600 s, the wind vector length will be 12000 m or 12 km. Thus, when the wind vector is plotted in the plot frame, it will appear to have the same length as the distance between two geographic features that are separated by 12 km. The user will often want to change **DTVEC** when changing the size of the plot window (see VALUE 8 under the **FLDNAME** entry below). Used with **HEADSPEED** when **FLDNAME** VALUE 4 Option 4 is U, V, or v.
HEADSPEED real	Speed, in m/sec. **HEADSPEED** controls the length of the plotted arrowhead.

	Length of the plotted arrowhead = **HEADSPEED** * **DTVEC**, and refers to a geographic distance in the same way as the vector length described for **DTVEC**. Used with **DTVEC** when **FLDNAME** Value 4 Option 4 is U, V, or v. A suggested value for **HEADSPEED** is approximately 10% of the maximum wind speed expected in the plot. 3 m/sec works well in many situations.
STEMLENGTH real	Distance, in m. **STEMLENGTH** controls the length of the stem of the plotted windbarbs. It refers to a geographic distance in the same way as the vector length described for **DTVEC**. Used with **DTVEC** when **FLDNAME** Value 4 Option 4 is B.
VEC_MAXMRL integer	The maximum mesh refinement level at which velocity vectors or windbarbs are plotted. If local mesh refinement is used, setting VEC_MAXMRL to a value lower than the total number of mesh refinement levels reduces the number of vectors (or windbarbs) plotted and helps prevent overcrowding of vectors on the finest grid regions. Used with **FLDNAME** Value 4 Option 4.
PRTVAL_SIZE character	For plots that display grid cell index numbers and/or field value numbers directly on the plot (see **PLOTSPECS:** VALUE 4, Options 2 & 3), specifies size of these numbers. **PRTVAL_SIZE** may be set to 'small', 'medium', or 'large'
PLTTYPE integer	Specifies graphical output file format (see **NPLT**). **PLTTYPE** = 0, NCAR graphic meta file **PLTTYPE** = 1, postscript file **PLTTYPE** = 2, pdf file (requires NCAR graphics 4.4.0 or later) **PLTTYPE** = 1 or 2 is best for **RUNTYPE** = 'PLOTONLY'
PLTNAME character	Output filename for graphical plots (see **NPLT**) For **PLTTYPE** = 1, .ps will be appended to **PLTNAME**

	For **PLTTYPE** = 2, .pdf will be appended to **PLTNAME**
PLTORIENT integer	Orientation of plotted output (see **NPLT and PLTTYPE**). Only used when **PLTTYPE** = 1 or 2. **PLTORIENT** = 0, portrait **PLTORIENT** = 1, landscape
ZPLOT_MIN real **ZPLOT_MAX** real	For vertical cross section (or cone) plots, minimum and maximum heights (m) of plot window. Special values of -1.0 for each parameter are flags indicating lowest and highest heights in the domain, respectively. That is, setting both parameters to -1.0 results in no reduced plot window.
PLOTSPECS data structure array	A data structure array used for specifying how fields are to be plotted. **PLOTSPECS** contains nine data members, all of which are entered on a single line in OLAMIN. These values must be separated by commas, and a comma is also required at the end of each line to separate the ninth data value in the line from the first data value in the subsequent line. The nine data members provide instructions for plotting a single data field. Separate lines are used for each field to be plotted. The number of lines actually used to generate plots is the number specified for NPLT. Thus, if **NPLT** is, say, 10, only the first 10 lines of **PLOTSPECS** will be used. Plots may be made when **RUNTYPE** = 'INITIAL', 'HISTORY', or 'PLOTONLY' The nine data values specified for each plotted field are commented in OLAMIN, and are as follows: VALUE 1. Character string. It is denoted as 'FLDNAME' in the OLAMIN comments, and indicates the name of the field to be plotted. There are currently over 300 fields to choose from. The available choices are listed in the file *oplot_lib.f90*. Additional fields are easily added to *oplot_lib.f90* by completing three steps: increase the parameter 'nfields' defined in the file; add a new field name character string, grid stagger point and field dimensionality, and character strings for plotted field name and units to the 'data fldlib' statement; and add the actual access of computation of the field value in a new 'case' section of the large 'IF' block in the file VALUE 2. Character. It is denoted as 'PROJECTN' in the OLAMIN comments, and indicates the plot projection.

When **MDOMAIN** = 0, options are:

'L' latitude-longitude projection of the earth onto the plotting plane

'P' polar stereographic projection of the earth onto the plotting plane

'O' orthographic projection of the earth onto the plotting plane

'C' vertical cross section for a global area simulation

The cross section is a conical surface with the cone vertex at the earth center, and the cone angle (from axis to surface) is chosen by the user anywhere between 0 and 180.

When **MDOMAIN** = 3, 4, or 5, options are:

'Z' horizontal Cartesian plot for a limited area simulation

'V' vertical cross section for a limited-area simulation

HOWEVER, 'C' SHOULD BE USED WITH **MDOMAIN** = 5.

VALUE 3. Integer. It is denoted by 'color table' in the OLAMIN comments and specifies the color table to be used to plot the field.

The color tables are located in file *colortables.f90*. Each color table consists of a list values and the colors that represent the values.

VALUE 4. Character string. It is denoted by 'PLOTSPEC2' in the OLAMIN comments and specifies plotting options.

There are 18 options to choose from, corresponding to the 18 lines of comments in OLAMIN listed for 'PLOTSPEC2'. Make 0 or 1 selection from each option. Note: Value 4 is a character string. Use quotes to enclose the list of option choices. Do not use commas between the option choices. Example: 'TJPGWtnm',

Option 1. Instructs how to fill in/use color.

T tile, each grid cell is filled with a single color

F color is filled in between contour lines without drawing the lines

L draws contour lines

O does both F and L

Option 2.

P print field value, valid only if Option 1 is T

Option 3.
I print grid cell index number, valid only if Option 1 is T
J print grid cell index number of all points in grid stagger

Option 4.
B print wind barbs.
V plot normal component of wind vector across V face of hexagonal grid cell.
w plot full horizontal wind vector defined at center of T grid cell.
For V or w, see **DTVEC** and **HEADSPEED.**
For B, see **STEMLENGTH.**

Option 5.
G plot atmospheric grid cells

Option 6.
g plot land/sea grid cells

Option 7.
D plot dual grid cells

Option 8.
b put a border/frame around plot
t put a border/frame around plot, put tick marks on the x and y axis, and label the tick marks

Option 9.
n print field name and units (from *oplot_lib.f90*) in upper left corner above plot
i same as n and adds extra info in lower right corner below plot

Option 10.
c plot color bar

Option 11.
M plot color-filled map (not yet functional)
m plot lined map showing land, country, and state borders

Option 12.
C plot path of cone circle of first field in **FLDNAME** list. This

option only applies for a plot with VALUE 2 set to 'L', 'P', or 'O', and requires that the first field in **FLDNAME** is actually plotted with a cone plot.

Option 13.

p plot on constant pressure surface

s plot near-surface value

Option 14.

W specifies that the plot window will include only a subset of the domain

Option 15.

e denotes that this field is to be 'external'. This means that it will not be plotted at the usual time when the other fields in this list are plotted, but instead only when specifically instructed from user-chosen locations within the code. This allows fields to be plotted that reside in temporary arrays rather than in permanent model arrays.

Option 16.

f when used, do not advance to next frame (no new page), and the next field is plotted over this one

Option 17.

Position of the plot in the frame, that is, position on the page.

Allows more than one plot per page

1 upper left

2 upper right

3 lower left

4 lower right

If Option 16, f, is used (no new page) and Option 17 is not, then a full page plot is made and the next field plot overlays this one.

Option 18.

u Do not mask out underground cells (with solid gray color)

Option 19.

N no extras

	VALUE 5. Real Specifies the longitude of the geographic center of the plot window VALUE 6. Real Specifies the latitude of the geographic center of the plot window VALUE 7. Real Cone angle or height. If VALUE 1 is C, then VALUE 7 is the cone angle (angle from cone axis to cone surface) and VALUE 7 can be > 0 to < 180 degrees. If VALUE 1 is L ,P, O, or Z, then VALUE 7 is the height above sea level, in meters, of the plot horizontal cross section. If VALUE 1 is V, VALUE 7 is ignored. VALUE 8. Real Specifies the plot window width in meters. The same value is used for width and height. VALUE 8 is only used if W is specified in VALUE 4. VALUE 9. Real Only used if Value 1 is C or V. VALUE 9 is the horizontal direction, in degrees azimuth (0 is north, 90 is east) of the view direction, that is, the direction that one is looking normal to the vertical cross section plot. This value determines the azimuthal direction of the vertical cross section, as well as the selection of which direction corresponds to right or left in the plot window.