

Interpola

v2.0

Generated by Doxygen 1.8.18

1 Interpola	1
2 Modules Index	3
2.1 Modules List	3
3 File Index	5
3.1 File List	5
4 Module Documentation	7
4.1 vars_dat Module Reference	7
4.1.1 Detailed Description	10
4.1.2 Variable Documentation	10
4.1.2.1 cday	10
4.1.2.2 cenlat	10
4.1.2.3 cenlon	11
4.1.2.4 cname	11
4.1.2.5 cunits	11
4.1.2.6 current_date	11
4.1.2.7 dix	11
4.1.2.8 djx	11
4.1.2.9 dlat	12
4.1.2.10 dlon	12
4.1.2.11 dpob	12
4.1.2.12 dx	12
4.1.2.13 dxe	12
4.1.2.14 dy	12
4.1.2.15 dye	13
4.1.2.16 ed	13
4.1.2.17 ei	13
4.1.2.18 eix	13
4.1.2.19 ejx	13
4.1.2.20 elat	13
4.1.2.21 elon	14
4.1.2.22 ename	14
4.1.2.23 epob	14
4.1.2.24 gmt	14
4.1.2.25 grid_id	14
4.1.2.26 isice	14
4.1.2.27 islake	15
4.1.2.28 isoilwater	15

4.1.2.29 isurban	15
4.1.2.30 iswater	15
4.1.2.31 itime	15
4.1.2.32 julday	15
4.1.2.33 julyr	16
4.1.2.34 map_proj_char	16
4.1.2.35 mapproj	16
4.1.2.36 mecha	16
4.1.2.37 mminlu	16
4.1.2.38 moadcenlat	16
4.1.2.39 ndims	17
4.1.2.40 nh	17
4.1.2.41 num_land_cat	17
4.1.2.42 pollat	17
4.1.2.43 pollon	17
4.1.2.44 radm	17
4.1.2.45 sdim	18
4.1.2.46 stdlon	18
4.1.2.47 times	18
4.1.2.48 title	18
4.1.2.49 tpob	18
4.1.2.50 trulat1	18
4.1.2.51 trulat2	19
4.1.2.52 tvar	19
4.1.2.53 unlimdimid	19
4.1.2.54 xlat	19
4.1.2.55 xlon	19
4.1.2.56 zlev	19
5 File Documentation	21
5.1 calculos.F90 File Reference	21
5.1.1 Function/Subroutine Documentation	21
5.1.1.1 conversion()	21
5.2 Interpola.F90 File Reference	22
5.2.1 Function/Subroutine Documentation	22
5.2.1.1 interpola()	22
5.3 lee_files.F90 File Reference	23
5.3.1 Function/Subroutine Documentation	23
5.3.1.1 check()	23

5.3.1.2 file_reading()	23
5.4 README.md File Reference	24
5.5 salidas.F90 File Reference	24
5.5.1 Function/Subroutine Documentation	24
5.5.1.1 crea_attr()	24
5.5.1.2 file_out()	25
5.6 vars_dat_mod.F90 File Reference	25
Index	29

Chapter 1

Interpola

Emission Interpolation to a new mesh by using a conservative flux method

Based on the wrfinput and wrfchem emissions inventory

input files:

```
wrfchemin.nc ! A 12 hours emission file to be interpolated (0 to 11 hour or 12 to 23 hour)
wrfinput      ! Domain where emissions will be interpolated
```

output file:

```
wrfchemi_00z_d01 or wrfchemi_12z_d01 ! 00z or 12z based on emissions file.
                                     ! d01, d02,... based on wrfinput
```

wrfchemin.nc - file contain emissions starting with "E_"



Chapter 2

Modules Index

2.1 Modules List

Here is a list of all modules with brief descriptions:

vars_dat	Set up variables used during the process	7
--------------------------	--	-------------------

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

calculos.F90	21
Interpola.F90	22
lee_files.F90	23
salidas.F90	24
vars_dat_mod.F90	25

Chapter 4

Module Documentation

4.1 vars_dat Module Reference

Set up variables used during the process.

Variables

- integer, parameter `nh` =24
- integer `ndims`
Number of dimension in wrfinput file.
- integer `zlev`
Number of emissions layers (1 to 8)
- integer `radm` =0
number of emissions classes
- real, dimension(:,:,:), allocatable `ei`
emissions input file dimensions `nx, ny, level, nh, radm`
- real, dimension(:,:,:), allocatable `ed`
emissions in new DOMAIN file dimensions `nx, ny, level, nh, radm`
- real, dimension(:,:), allocatable `elat`
Latitudes from input file emissions.
- real, dimension(:,:), allocatable `elon`
Longitudes from input file emissions.
- real, dimension(:,:), allocatable `epob`
Density population from input file emissions.
- real, dimension(:,:), allocatable `dlat`
Latitudes in new domain `nx, ny` from new domain `ed`.
- real, dimension(:,:), allocatable `dlon`
Longitudes in new domain `nx, ny` from new domain `ed`.
- real, dimension(:,:), allocatable `dpob`
Density population in new domain `nx, ny` from new domain `ed`.
- real, dimension(:,:), allocatable `xlon`

- *Longitudes in emissions domain nx, ny .*
- real, dimension(:, :, :), allocatable [xlat](#)
- *Latitudes in emissions domain nx, ny .*
- integer [dix](#)
- *Number of values in longitude in new file.*
- integer [djx](#)
- *Number of values in latitude in new file.*
- integer [eix](#)
- *Number of values in longitude in emissions file.*
- integer [ejx](#)
- *Number of values in latitude in emissions file.*
- integer [grid_id](#)
- *Domain number (d01, d02, etc.) from wrfinput.*
- integer [julyr](#)
- *Julian year in emissions file.*
- integer [julday](#)
- *Julian day in emissions file.*
- integer [mapproj](#)
- *Map projection type.*
- integer [iswater](#)
- *Value for land use water.*
- integer [islake](#)
- *Value for land use lake.*
- integer [isice](#)
- *Value for land use ice.*
- integer [isurban](#)
- *Value for land use urban.*
- integer [isoilwater](#)
- *Value for land use ice.*
- integer [unlimdimid](#)
- *ID unlimit variable (time)*
- real [cenlat](#)
- *Central latitude.*
- real [cenlon](#)
- *Central longitude.*
- real [dx](#)
- *Grid dimension in m output file x.*
- real [dy](#)
- *Grid dimension in m output file y.*
- real [dx](#)
- *Grid dimension in m emissions file x.*
- real [dye](#)
- *Grid dimension in m emissions file y.*
- real [trulat1](#)
- *True latitud lower.*
- real [trulat2](#)
- *True latitud higer.*

- real [moadcenlat](#)
Mother of all domains center latitude.
- real [stdlon](#)
Standard longitude.
- real [pollat](#)
The pole latitude.
- real [pollon](#)
The pole longitude.
- real [gmt](#)
GMT time.
- real [num_land_cat](#)
Number of land categories.
- character(len=3) [cday](#)
Day type (lun, mar, mie, jue, vie, sab, dom)
- character(len=19) [mminlu](#)
Land use input description.
- character(len=19) [map_proj_char](#)
Map projection description.
- character(len=19) [itime](#)
Counter for time in file.
- character(len=38) [title](#)
Title description input/output files for V4 should have V4.0.
- character(len=19), dimension(1, 1) [times](#)
Start date in input emissions file.
- character(len=19) [current_date](#)
Current date in input emissions file.
- character(len=19) [mecha](#)
Chemical mechanism name.
- character(len=19), dimension(:), allocatable [sdim](#)
Vector of dimensions descriptions.
- character(len=11), dimension(:), allocatable [ename](#)
Emissions description long.
- character(len=50), dimension(:), allocatable [cname](#)
Emissions name variable short.
- character(len=50), dimension(:), allocatable [cunits](#)
Units in emissions vars.
- logical, dimension(:), allocatable [tvar](#)
true if input var is an emissions variable
- logical [tpob](#)
true if input emissions files contains density population

4.1.1 Detailed Description

Set up variables used during the process.

Author

Jose Agustin Garcia Reynoso

Date

28-08-2012

Emissions Inventories Variables

Parameters

<i>nh</i>	Number of hours during the day
-----------	--------------------------------

4.1.2 Variable Documentation

4.1.2.1 cday

```
character(len=3) vars_dat::cday
```

Day type (lun, mar, mie, jue, vie, sab, dom)

4.1.2.2 cenlat

```
real vars_dat::cenlat
```

Central latitude.

4.1.2.3 cenlon

```
real vars_dat::cenlon
```

Central longitude.

4.1.2.4 cname

```
character(len=50), dimension(:), allocatable vars_dat::cname
```

Emissions name variable short.

4.1.2.5 cunits

```
character(len=50), dimension(:), allocatable vars_dat::cunits
```

Units in emissions vars.

4.1.2.6 current_date

```
character (len=19) vars_dat::current_date
```

Current date in input emissions file.

4.1.2.7 dix

```
integer vars_dat::dix
```

Number of values in longitude in new file.

4.1.2.8 djx

```
integer vars_dat::djx
```

Number of values in latitude in new file.

4.1.2.9 dlat

```
real, dimension(:,:), allocatable vars_dat::dlat
```

Latitudes in new domain nx, ny from new domain ed .

4.1.2.10 dlon

```
real, dimension(:,:), allocatable vars_dat::dlon
```

Longitudes in new domain nx, ny from new domain ed .

4.1.2.11 dpob

```
real, dimension(:,:), allocatable vars_dat::dpob
```

Density population in new domain nx, ny from new domain ed .

4.1.2.12 dx

```
real vars_dat::dx
```

Grid dimension in m output file x .

4.1.2.13 dxe

```
real vars_dat::dxe
```

Grid dimension in m emissions file x .

4.1.2.14 dy

```
real vars_dat::dy
```

Grid dimension in m output file y .

4.1.2.15 dye

```
real vars_dat::dye
```

Grid dimension in m emissions file y.

4.1.2.16 ed

```
real, dimension(:,:,:,:), allocatable vars_dat::ed
```

emissions in new DOMAIN file dimensions nx, ny, level, nh, radm

4.1.2.17 ei

```
real, dimension(:,:,:,:), allocatable vars_dat::ei
```

emissions input file dimensions nx, ny, level, nh, radm

4.1.2.18 eix

```
integer vars_dat::eix
```

Number of values in longitude in emissions file.

4.1.2.19 ejx

```
integer vars_dat::ejx
```

Number of values in latitude in emissions file.

4.1.2.20 elat

```
real, dimension(:,:), allocatable vars_dat::elat
```

Latitudes from input file emissions.

4.1.2.21 elon

```
real, dimension(:,:), allocatable vars_dat::elon
```

Longitudes from input file emissions.

4.1.2.22 ename

```
character(len=11), dimension(:), allocatable vars_dat::ename
```

Emissions description long.

4.1.2.23 epob

```
real, dimension(:,:), allocatable vars_dat::epob
```

Density population from input file emissions.

4.1.2.24 gmt

```
real vars_dat::gmt
```

GMT time.

4.1.2.25 grid_id

```
integer vars_dat::grid_id
```

Domain number (d01, d02, etc.) from wrfinput.

4.1.2.26 isice

```
integer vars_dat::isice
```

Value for land use ice.

4.1.2.27 islake

```
integer vars_dat::islake
```

Value for land use lake.

4.1.2.28 isoilwater

```
integer vars_dat::isoilwater
```

Value for land use ice.

4.1.2.29 isurban

```
integer vars_dat::isurban
```

Value for land use urban.

4.1.2.30 iswater

```
integer vars_dat::iswater
```

Value for land use water.

4.1.2.31 itime

```
character(len=19) vars_dat::itime
```

Counter for time in file.

4.1.2.32 julday

```
integer vars_dat::julday
```

Julian day in emissions file.

4.1.2.33 `julyr`

```
integer vars_dat::julyr
```

Julian year in emissions file.

4.1.2.34 `map_proj_char`

```
character(len=19) vars_dat::map_proj_char
```

Map projection description.

4.1.2.35 `mapproj`

```
integer vars_dat::mapproj
```

Map projection type.

4.1.2.36 `mecha`

```
character (len=19) vars_dat::mecha
```

Chemical mechanism name.

4.1.2.37 `mminlu`

```
character(len=19) vars_dat::mminlu
```

Land use input description.

4.1.2.38 `moadcenlat`

```
real vars_dat::moadcenlat
```

Mother of all domains center latitude.

4.1.2.39 ndims

```
integer vars_dat::ndims
```

Number of dimension in wrfinput file.

4.1.2.40 nh

```
integer, parameter vars_dat::nh =24
```

4.1.2.41 num_land_cat

```
real vars_dat::num_land_cat
```

Number of land categories.

4.1.2.42 pollat

```
real vars_dat::pollat
```

The pole latitude.

4.1.2.43 pollon

```
real vars_dat::pollon
```

The pole longitude.

4.1.2.44 radm

```
integer vars_dat::radm =0
```

number of emissions classes

4.1.2.45 sdim

```
character (len=19), dimension(:), allocatable vars_dat::sdim
```

Vector of dimensions descriptions.

4.1.2.46 stdlon

```
real vars_dat::stdlon
```

Standard longitude.

4.1.2.47 times

```
character(len=19), dimension(1,1) vars_dat::times
```

Start date in input emissions file.

4.1.2.48 title

```
character(len=38) vars_dat::title
```

Title description input/output files for V4 should have V4.0.

4.1.2.49 tpob

```
logical vars_dat::tpob
```

true if input emissions files contains density population

4.1.2.50 trulat1

```
real vars_dat::trulat1
```

True latitud lower.

4.1.2.51 trulat2

```
real vars_dat::trulat2
```

True latitud higer.

4.1.2.52 tvar

```
logical, dimension(:), allocatable vars_dat::tvar
```

true if input var is an emissions variable

4.1.2.53 unlimdimid

```
integer vars_dat::unlimdimid
```

ID unlimit variable (time)

4.1.2.54 xlat

```
real, dimension(:,:,:), allocatable vars_dat::xlat
```

Latitudes in emissions domain nx, ny.

4.1.2.55 xlon

```
real, dimension(:,:,:), allocatable vars_dat::xlon
```

Longitudes in emissions domain nx, ny.

4.1.2.56 zlev

```
integer vars_dat::zlev
```

Number of emissions layers (1 to 8)

Chapter 5

File Documentation

5.1 calculos.F90 File Reference

Functions/Subroutines

- subroutine [conversion](#)

It does the interpolation into the new Mesh.

5.1.1 Function/Subroutine Documentation

5.1.1.1 conversion()

```
subroutine conversion
```

It does the interpolation into the new Mesh.

Interpolates the emissions into new mesh conserving mass uses emission area and the fractional area between the original and new grid to set the emissions.

Computes the mass in the original mesh and compares against the new mesh, if both domains cover the same area the ratio `xemis/ xmas` should be 1

Author

Jose Agustin Garcia Reynoso

Date

28/08/2012.

Version

2.0

5.2 Interpola.F90 File Reference

Functions/Subroutines

- program [interpola](#)

Emission Interpolation from one mesh to a new mesh.

5.2.1 Function/Subroutine Documentation

5.2.1.1 `interpola()`

```
program interpola
```

Emission Interpolation from one mesh to a new mesh.

Contains a call for tree subroutines that completes the procedure

`file_reading`

Reads Emission inventory and the mesh to interpolate.

`conversion`

Computations for emissions mass conservation into the new mesh.

`file_out`

Create output file and write results

Author

Jose Agustin Garcia Reynoso

Date

2012/06/20

Version

2.0

Copyright

Universidad Nacional Autonoma de Mexico.

5.3 lee_files.F90 File Reference

Functions/Subroutines

- subroutine [file_reading](#)
Reads Emission inventory and the new Mesh to interpolate emissions.
- subroutine [check](#) (status)
Evaluation of netcdf status.

5.3.1 Function/Subroutine Documentation

5.3.1.1 check()

```
subroutine check (
    integer, intent(in) status )
```

Evaluation of netcdf status.

In case of error prints error message description

Parameters

<i>status</i>	An error status that might have been returned from a previous call to some netCDF function
---------------	--

Date

28/08/2012.

5.3.1.2 file_reading()

```
subroutine file_reading
```

Reads Emission inventory and the new Mesh to interpolate emissions.

Reads from the emission wrfchemin file the variables and attributes put emissions in `ei` array and coordinates in `xlat`, `xlon`.

reads the new mesh from wrfinput, stores the new coordinates `dlat`, `dlon`

Author

Agustin Garcia

Date

28/08/2012.

Version

2.0

Copyright

Universidad Nacional Autonoma de Mexico.

5.4 README.md File Reference

5.5 salidas.F90 File Reference

Functions/Subroutines

- subroutine [file_out](#)
file_out creates the output file and writes the interpolated emissions from the new mesh
- subroutine [crea_attr](#) (ncid, idm, dimids, svar, cname, cunits, id_var)
creates attributes for gas variables and aerosol variables

5.5.1 Function/Subroutine Documentation

5.5.1.1 crea_attr()

```
subroutine file_out::crea_attr (  
    integer, intent(in) ncid,  
    integer, intent(in) idm,  
    integer, dimension(idm), intent(in) dimids,  
    character(len=*), intent(in) svar,  
    character(len=*), intent(in) cname,  
    character(len=*), intent(in) cunits,  
    integer, intent(out) id_var )
```

creates attributes for gas variables and aerosol variables

Author

Agustin Garcia

Date

28/08/2012

Parameters

in	<i>ncid</i>	netCDF ID, from a previous call to NF90_OPEN or NF90_CREATE
in	<i>idm</i>	Number of dimensions in <i>dimids</i>
out	<i>id_var</i>	ID from variable to store
in	<i>dimids</i>	Array with ID for each dimension
in	<i>svar</i>	Short name of variable to store
in	<i>cname</i>	Description of variable to store
in	<i>cunits</i>	Units for variable to store

5.5.1.2 file_out()

```
subroutine file_out
```

file_out creates the output file and writes the interpolated emissions from the new mesh

Uses the attributes from wrfinpout file

Uses current_date from wrfchemin file

Author

Agustin Garcia

Date

28/08/2012

5.6 vars_dat_mod.F90 File Reference**Modules**

- module [vars_dat](#)

Set up variables used during the process.

Variables

- integer, parameter `vars_dat::nh` =24
- integer `vars_dat::ndims`
Number of dimension in wrfinput file.
- integer `vars_dat::zlev`
Number of emissions layers (1 to 8)
- integer `vars_dat::radm` =0
number of emissions classes
- real, dimension(:, :, :, :), allocatable `vars_dat::ei`
emissions input file dimensions nx, ny, level, nh, radm
- real, dimension(:, :, :, :), allocatable `vars_dat::ed`
emissions in new DOMAIN file dimensions nx, ny, level, nh, radm
- real, dimension(:, :), allocatable `vars_dat::elat`
Latitudes from input file emissions.
- real, dimension(:, :), allocatable `vars_dat::elon`
Longitudes from input file emissions.
- real, dimension(:, :), allocatable `vars_dat::epob`
Density population from input file emissions.
- real, dimension(:, :), allocatable `vars_dat::dlat`
Latitudes in new domain nx, ny from new domain ed.
- real, dimension(:, :), allocatable `vars_dat::dlon`
Longitudes in new domain nx, ny from new domain ed.
- real, dimension(:, :), allocatable `vars_dat::dpob`
Density population in new domain nx, ny from new domain ed.
- real, dimension(:, :, :), allocatable `vars_dat::xlon`
Longitudes in emissions domain nx, ny.
- real, dimension(:, :, :), allocatable `vars_dat::xlat`
Latitudes in emissions domain nx, ny.
- integer `vars_dat::dix`
Number of values in longitude in new file.
- integer `vars_dat::djx`
Number of values in latitude in new file.
- integer `vars_dat::eix`
Number of values in longitude in emissions file.
- integer `vars_dat::ejx`
Number of values in latitude in emissions file.
- integer `vars_dat::grid_id`
Domain number (d01, d02, etc.) from wrfinput.
- integer `vars_dat::julyr`
Julian year in emissions file.
- integer `vars_dat::julday`
Julian day in emissions file.
- integer `vars_dat::mapproj`
Map projection type.
- integer `vars_dat::iswater`
Value for land use water.

- integer `vars_dat::islake`
Value for land use lake.
- integer `vars_dat::isice`
Value for land use ice.
- integer `vars_dat::isurban`
Value for land use urban.
- integer `vars_dat::isoilwater`
Value for land use ice.
- integer `vars_dat::unlimdimid`
ID unlimit variable (time)
- real `vars_dat::cenlat`
Central latitude.
- real `vars_dat::cenlon`
Central longitude.
- real `vars_dat::dx`
Grid dimension in m output file x.
- real `vars_dat::dy`
Grid dimension in m output file y.
- real `vars_dat::dxo`
Grid dimension in m emissions file x.
- real `vars_dat::dyo`
Grid dimension in m emissions file y.
- real `vars_dat::trulat1`
True latitud lower.
- real `vars_dat::trulat2`
True latitud higer.
- real `vars_dat::moadcenlat`
Mother of all domains center latitude.
- real `vars_dat::stdlon`
Standard longitude.
- real `vars_dat::pollat`
The pole latitude.
- real `vars_dat::pollon`
The pole longitude.
- real `vars_dat::gmt`
GMT time.
- real `vars_dat::num_land_cat`
Number of land categories.
- character(len=3) `vars_dat::cday`
Day type (lun, mar, mie, jue, vie, sab, dom)
- character(len=19) `vars_dat::mminlu`
Land use input description.
- character(len=19) `vars_dat::map_proj_char`
Map projection description.
- character(len=19) `vars_dat::itime`
Counter for time in file.
- character(len=38) `vars_dat::title`

Title description input/output files for V4 should have V4.0.

- character(len=19), dimension(1, 1) [vars_dat::times](#)

Start date in input emissions file.

- character(len=19) [vars_dat::current_date](#)

Current date in input emissions file.

- character(len=19) [vars_dat::mecha](#)

Chemical mechanism name.

- character(len=19), dimension(:), allocatable [vars_dat::sdim](#)

Vector of dimensions descriptions.

- character(len=11), dimension(:), allocatable [vars_dat::ename](#)

Emissions description long.

- character(len=50), dimension(:), allocatable [vars_dat::cname](#)

Emissions name variable short.

- character(len=50), dimension(:), allocatable [vars_dat::cunits](#)

Units in emissions vars.

- logical, dimension(:), allocatable [vars_dat::tvar](#)

true if input var is an emissions variable

- logical [vars_dat::tpob](#)

true if input emissions files contains density population

Index

calculos.F90, [21](#)
 conversion, [21](#)
cday
 vars_dat, [10](#)
cenlat
 vars_dat, [10](#)
cenlon
 vars_dat, [10](#)
check
 lee_files.F90, [23](#)
cname
 vars_dat, [11](#)
conversion
 calculos.F90, [21](#)
crea_attr
 salidas.F90, [24](#)
cunits
 vars_dat, [11](#)
current_date
 vars_dat, [11](#)

dix
 vars_dat, [11](#)
dix
 vars_dat, [11](#)
dlat
 vars_dat, [11](#)
dlon
 vars_dat, [12](#)
dpob
 vars_dat, [12](#)
dx
 vars_dat, [12](#)
dx
 vars_dat, [12](#)
dye
 vars_dat, [12](#)
dye
 vars_dat, [12](#)

ed
 vars_dat, [13](#)
ei
 vars_dat, [13](#)
eix
 vars_dat, [13](#)

ejx
 vars_dat, [13](#)
elat
 vars_dat, [13](#)
elon
 vars_dat, [13](#)
ename
 vars_dat, [14](#)
epob
 vars_dat, [14](#)

file_out
 salidas.F90, [25](#)
file_reading
 lee_files.F90, [23](#)

gmt
 vars_dat, [14](#)
grid_id
 vars_dat, [14](#)

interpola
 Interpola.F90, [22](#)
Interpola.F90, [22](#)
 interpola, [22](#)
isice
 vars_dat, [14](#)
islake
 vars_dat, [14](#)
isoilwater
 vars_dat, [15](#)
isurban
 vars_dat, [15](#)
iswater
 vars_dat, [15](#)
itime
 vars_dat, [15](#)

julday
 vars_dat, [15](#)
julyr
 vars_dat, [15](#)

lee_files.F90, [23](#)
 check, [23](#)
 file_reading, [23](#)

map_proj_char
 vars_dat, 16
 mapproj
 vars_dat, 16
 mecha
 vars_dat, 16
 mminlu
 vars_dat, 16
 moadcenlat
 vars_dat, 16

 ndims
 vars_dat, 16
 nh
 vars_dat, 17
 num_land_cat
 vars_dat, 17

 pollat
 vars_dat, 17
 pollon
 vars_dat, 17

 radm
 vars_dat, 17
 README.md, 24

 salidas.F90, 24
 crea_attr, 24
 file_out, 25
 sdim
 vars_dat, 17
 stdlon
 vars_dat, 18

 times
 vars_dat, 18
 title
 vars_dat, 18
 tpob
 vars_dat, 18
 trulat1
 vars_dat, 18
 trulat2
 vars_dat, 18
 tvar
 vars_dat, 19

 unlimdimid
 vars_dat, 19

 vars_dat, 7
 cday, 10
 cenlat, 10
 cenlon, 10
 cname, 11
 cunits, 11
 current_date, 11
 dix, 11
 dix, 11
 dlat, 11
 dlon, 12
 dpob, 12
 dx, 12
 dx, 12
 dy, 12
 dye, 12
 ed, 13
 ei, 13
 eix, 13
 ejx, 13
 elat, 13
 elon, 13
 ename, 14
 epob, 14
 gmt, 14
 grid_id, 14
 isice, 14
 islake, 14
 isoilwater, 15
 isurban, 15
 iswater, 15
 itime, 15
 julday, 15
 julyr, 15
 map_proj_char, 16
 mapproj, 16
 mecha, 16
 mminlu, 16
 moadcenlat, 16
 ndims, 16
 nh, 17
 num_land_cat, 17
 pollat, 17
 pollon, 17
 radm, 17
 sdim, 17
 stdlon, 18
 times, 18
 title, 18
 tpob, 18
 trulat1, 18
 trulat2, 18
 tvar, 19
 unlimdimid, 19
 xlat, 19
 xlon, 19
 zlev, 19
 vars_dat_mod.F90, 25

xlat
 vars_dat, [19](#)
xlon
 vars_dat, [19](#)
zlev
 vars_dat, [19](#)