

DCPAM Reference Manual

GFD Dennou Club

平成 17 年 2 月 18 日

目次

1	Module type_mod	2
1.1	Overview	2
1.2	Error Handling	2
1.3	Known Bugs	2
1.4	Note	2
1.5	Future Plans	2
1.6	Dependency	2
1.7	Public Interface	2
1.8	Public Data	2
2	Module dc_message	3
2.1	Overview	3
2.2	Error Handling	3
2.3	Known Bugs	3
2.4	Note	3
2.5	Future Plans	3
2.6	Public Interface	3
2.7	Generic Procedure	3
2.8	Procedure Interface	4
2.8.1	メッセージ・警告・エラーの出力	4
2.8.2	メッセージ・警告・エラーの出力	4
2.9	Output Example	5
3	Module nmlfile_mod	5
3.1	Overview	5
3.2	Error Handling	6
3.3	Known Bugs	6
3.4	Note	6
3.5	Future Plans	6
3.6	Dependency	6
3.7	Public Interface	6
3.8	Procedure Interface	6
3.8.1	Initialize module and acquire NAMELIST	6
3.8.2	Open NAMELIST file, and Return Device Number	7
3.8.3	Close NAMELIST file	7
3.8.4	Return NAMELIST file name	8
3.8.5	Terminate module	8

4	Module constants_mod	8
4.1	Overview	9
4.2	Error Handling	9
4.3	Known Bugs	9
4.4	Note	9
4.5	Future Plans	9
4.6	Dependency	9
4.7	Public Interface	9
4.8	Public Data	9
4.9	Procedure Interface	10
4.9.1	Initialize module and acquire NAMELIST	10
4.9.2	Terminate module	11
5	Module time_mod	11
5.1	Overview	12
5.2	Error Handling	12
5.3	Known Bugs	12
5.4	Note	12
5.5	Future Plans	12
5.6	Dependency	12
5.7	Public Interface	12
5.8	Public Data	12
5.9	Procedure Interface	13
5.9.1	Initialize module and acquire NAMELIST	13
5.9.2	Progress Time and Number of Loop	14
5.9.3	Terminate module	14
6	Module grid_3d_mod	14
6.1	Overview	15
6.2	Error Handling	15
6.3	Known Bugs	15
6.4	Note	15
6.5	Future Plans	15
6.6	Dependency	15
6.7	Public Interface	15
6.8	Public Data	15
6.9	Procedure Interface	15
6.9.1	Initialize module and acquire NAMELIST	15
6.9.2	Terminate module	16

7	Module <code>grid_wavenumber_mod</code>	16
7.1	Overview	17
7.2	Error Handling	17
7.3	Known Bugs	17
7.4	Note	17
7.5	Future Plans	17
7.6	Dependency	17
7.7	Public Interface	17
7.8	Public Data	17
7.9	Procedure Interface	17
7.9.1	Initialize module and acquire NAMELIST	17
7.9.2	Terminate module	18
8	Module <code>spml_mod</code>	18
8.1	Overview	18
8.2	Error Handling	19
8.3	Known Bugs	19
8.4	Note	19
8.5	Future Plans	19
8.6	Dependency	19
8.7	Public Interface	19
8.8	Procedure Interface	19
8.8.1	Initialize <code>wa_module</code>	19
8.8.2	Terminate module	20
9	Module <code>axis_type_mod</code>	20
9.1	Overview	20
9.2	Error Handling	20
9.3	Known Bugs	20
9.4	Note	20
9.5	Future Plans	20
9.6	Dependency	20
9.7	Public Interface	21
9.8	Generic Procedure	21
9.9	Derived Types	21
9.10	Procedure Interface	21
9.10.1	Copy <code>AXISINFO</code> data (for 0 dimensional data) . . .	21
9.10.2	Copy <code>AXISINFO</code> data (for 1 dimensional data) . . .	22
9.10.3	Copy <code>AXISATTR</code> (in <code>gt4f90io</code>) data (for 1 dimensional data)	22

9.10.4	Initialize AXISATTR (in gt4f90io) data (for 0 dimensional data)	22
9.10.5	Initialize AXISATTR (in gt4f90io) data (for 1 dimensional data)	23
10	Module axis_x_mod	23
10.1	Overview	23
10.2	Error Handling	24
10.3	Known Bugs	24
10.4	Note	24
10.5	Future Plans	24
10.6	Dependency	24
10.7	Public Interface	24
10.8	Procedure Interface	24
10.8.1	Initialize module and acquire NAMELIST	24
10.8.2	Return Weight of axis X	27
10.8.3	Return Data of axis X for Spectral Method.	27
10.8.4	Return Data of axis X from NAMELIST	28
10.8.5	Return Data of axis X from netCDF file	28
10.8.6	Terminate module	29
11	Module axis_y_mod	29
11.1	Overview	29
11.2	Error Handling	29
11.3	Known Bugs	29
11.4	Note	30
11.5	Future Plans	30
11.6	Dependency	30
11.7	Public Interface	30
11.8	Procedure Interface	30
11.8.1	Initialize module and acquire NAMELIST	30
11.8.2	Return Weight of axis Y	32
11.8.3	Return Data of axis Y for Spectral Method.	33
11.8.4	Return Data of axis Y from NAMELIST	34
11.8.5	Return Data of axis Y from netCDF file	34
11.8.6	Terminate module	35
12	Module axis_z_mod	35
12.1	Overview	35
12.2	Error Handling	35

12.3	Known Bugs	35
12.4	Note	35
12.5	Future Plans	35
12.6	Dependency	35
12.7	Public Interface	36
12.8	Procedure Interface	36
12.8.1	Initialize module and acquire NAMELIST	36
12.8.2	Return Sigma Full Level and Sigma Half Level (from NAMELIST) as Data of axis Z	40
12.8.3	Return Sigma Full Level and Sigma Half Level (from netCDF Data) as Data of axis Z	41
12.8.4	Return Data of axis Z from NAMELIST	41
12.8.5	Return Data of axis Z from netCDF file	41
12.8.6	Return Half Level Data of axis Z from NAMELIST	42
12.8.7	Return Half Level Data of axis Z from netCDF file	42
12.8.8	Generate Sigma Level from Sigma Half Level.	43
12.8.9	Terminate module	43
13	Module varinfo_mod	44
13.1	Overview	44
13.2	Error Handling	44
13.3	Known Bugs	44
13.4	Note	44
13.5	Future Plans	44
13.6	Dependency	44
13.7	Public Interface	44
13.8	Generic Procedure	45
13.9	Derived Types	45
13.10	Procedure Interface	45
13.10.1	Initialize module and acquire NAMELIST	45
13.10.2	Return Variable Information about varkey.	48
13.10.3	Terminate module	48
13.10.4	Copy VAR_INFO array.	48
13.10.5	Copy GT_HISTORY_ATTR array.	49
13.10.6	Initialize GT_HISTORY_ATTR variable.	49
13.10.7	Initialize GT_HISTORY_ATTR variable.	50
14	Module io_gt4_out_mod	50
14.1	Overview	50
14.2	Error Handling	50

14.3	Known Bugs	50
14.4	Note	51
14.5	Future Plans	51
14.6	Dependency	51
14.7	Public Interface	51
14.8	Generic Procedure	51
14.9	Derived Types	51
14.10	Procedure Interface	52
14.10.1	Initialize module and acquire NAMELIST	52
14.10.2	Set Dimension	53
14.10.3	Set Variables	53
14.10.4	Put 3-Dimensional Single Precision Data to netCDF file	54
14.10.5	Put 2-Dimensional Single Precision Data to netCDF file	55
14.10.6	Put 0-Dimensional Single Precision Data to netCDF file	55
14.10.7	Put 3-Dimensional Double Precision Data to netCDF file	56
14.10.8	Put 2-Dimensional Double Precision Data to netCDF file	56
14.10.9	Put 2-Dimensional Double Precision Data to netCDF file	57
14.10.10	Terminate module	57
15	Module dynamics_mod	57
15.1	Overview	58
15.2	Reference	58
15.3	Error Handling	58
15.4	Known Bugs	58
15.5	Note	59
15.6	Future Plans	59
15.7	Dependency	59
15.8	Public Interface	59
15.9	Procedure Interface	59
15.9.1	Initialize module and Calculate Non-Predictional Value.	59
15.9.2	Calculate Predictional Values.	60
15.9.3	Calculate Diffusion Term	62
15.9.4	Calculate Diagnostic Values.	62
15.9.5	Terminate module	64

16	Module hs94forcing_mod	64
16.1	Overview	64
16.2	Reference	64
16.3	Error Handling	64
16.4	Known Bugs	64
16.5	Note	65
16.6	Future Plans	65
16.7	Dependency	65
16.8	Public Interface	65
16.8.1	加熱と散逸の計算	65
16.8.2	Terminate module	66

1 Module type_mod

- Developers: Morikawa Yasuhiro
- Version: \$Id: type.f90,v 1.4 2005/01/19 08:52:44 morikawa Exp \$
- Tag Name: \$Name: \$
- Change History:

1.1 Overview

This module provide parameter of KIND.

種別型パラメタ値を提供するモジュールである。この他のモジュールおよびプログラムは原則的に全てこのモジュールを参照するようにする。そのようにすることで、種別型パラメタ値を変更したい場合、このモジュール内の値を変更するだけで全てのモジュールにその変更が適用される。

1.2 Error Handling

1.3 Known Bugs

1.4 Note

1.5 Future Plans

1.6 Dependency

none.

1.7 Public Interface

```
private
public :: REKIND, DBKIND, INTKIND, TOKEN, STRING, NMLARRAY ! variables
```

1.8 Public Data

```
integer, parameter:: TOKEN = 32 ! 単語の文字列の長さ
integer, parameter:: STRING = 500 ! 文章の文字列の長さ
```

```
integer, parameter:: REKIND = kind(0.0e0) ! 単精度実数の精度
integer, parameter:: DBKIND = kind(0.0d0) ! 倍精度実数の精度
```

```
integer, parameter:: INTKIND = kind(0)      ! 整数の精度
```

```
integer, parameter:: NMLARRAY = 100 ! NAMELIST で配列を取得する  
際の
```

```
! デフォルトの配列サイズ
```

2 Module dc_message

- Editor: Morikawa Yasuhiro
- Last Update: 2004/12/25
- Created: 2004/12/23

2.1 Overview

エラー処理も含めたメッセージ・警告の出力を行なう。dcl の MSGDMP.f の上位互換としても利用できるようにしてある (つもり)。エラー処理自体は dc_error モジュールに依存。出力装置は標準出力に決め打ち。

2.2 Error Handling

2.3 Known Bugs

2.4 Note

2.5 Future Plans

2.6 Public Interface

```
private  
public:: MessageNotify ! subroutines
```

2.7 Generic Procedure

```
interface MessageNotify  
  module procedure MessageNotifyC  
  module procedure MessageNotifyI  
end interface
```

2.8 Procedure Interface

2.8.1 メッセージ・警告・エラーの出力

メッセージ・警告・エラーを出力したい場合に用います。標準出力に出力されます。

文字型変数 `where` にはプログラム名 (サブルーチン名) など、プログラム内のどこでメッセージを出力するのかが示すものを与えます。

文字型変数 `message` には、出力したい文字列を与えます。オプション変数 `i`, `r`, `d`, `L`, `s`, `n`, `c1`, `c2`, `c3` を付加する事も出来ます。詳細に関しては `dc_string` モジュールの `CPrintf` を参照して下さい。

文字型変数 `level` は出力するメッセージの種類を決める引数で、`'W'` (または `'Warning'` など `'W'` で始まる文字) を与える事で警告を、`'E'` (または `'Error'` など `'E'` で始まる文字) を与える事でエラー (メッセージ出力後プログラムを終了) を、それ以外の文字を与える事で通常のメッセージを出力します。なお、エラー出力する場合、エラーコードは `USR_ECHAR` となります。

```
subroutine MessageNotifyC(level, where, message, &
    & i, r, d, L, n, c1, c2, c3)
```

Dependency

```
use dc_types ,only: string
use dc_string ,only: UChar, StrHead, Printf, CPrintf
use dc_error ,only: StoreError, USR_ECHAR
```

Input

```
character(*), intent(in)      :: level
character(*), intent(in)      :: where
character(*), intent(in)      :: message
integer      , intent(in), optional:: i(:), n(:)
real         , intent(in), optional:: r(:)
real(8)      , intent(in), optional:: d(:)
logical      , intent(in), optional:: L(:)
character(*), intent(in), optional:: c1, c2, c3
```

2.8.2 メッセージ・警告・エラーの出力

原則的に `MessageNotifyC` と同様ですが、こちらは第 1 引数に数値型変数 `number` をとります。この `number` はそのまま `StoreError` に引き渡されます。

```
subroutine MessageNotifyI(number, where, message, &
    & i, r, d, L, n, c1, c2, c3)
```

Dependency

```
use dc_string ,only: CPrintf
use dc_error  ,only: StoreError, USR_ECHAR
```

Input

```
integer,      intent(in)           :: number
character(*), intent(in)           :: where
character(*), intent(in), optional:: message
integer      , intent(in), optional:: i(:), n(:)
real         , intent(in), optional:: r(:)
real(8)      , intent(in), optional:: d(:)
logical      , intent(in), optional:: L(:)
character(*), intent(in), optional:: c1, c2, c3
```

2.9 Output Example

```
*** MESSAGE [where] *** message

*** WARNING [where] *** message

*** ERROR (Code number) [where] *** message
```

3 Module nmlfile_mod

- Developers: Morikawa Yasuhiro
- Version: \$Id: nmlfile.f90,v 1.9 2005/01/19 08:52:37 morikawa Exp \$
- Tag Name: \$Name: \$
- Change History:

3.1 Overview

This module support to input NAMELIST.

NAMELIST を取得するための支援モジュールである。ファイルから NAMELIST を取得する場合に便利なサブルーチンを用意している。NAMELIST を取得するモジュールおよびプログラムは、原則的に全てこのモジュールを呼ぶ。

3.2 Error Handling

3.3 Known Bugs

3.4 Note

3.5 Future Plans

現在、あるプログラムで読み込む事が可能な NAMELIST ファイルは、メインプログラムで `nmlfile_init(nml)` と呼んだ際の `nml` というファイル名に固定されてしまう。実行プログラムで引数としてファイル名を取得したり、標準入力からファイル名を取得するなどして、再コンパイル無しに NAMELIST ファイルを変更できると良いかもしれない。

3.6 Dependency

```
use type_mod,    only : TOKEN, INTKIND, STRING
```

3.7 Public Interface

```
private
public :: nmlfile_init, nmlfile_end    ! subroutines
public :: nmlfile_open, nmlfile_close ! subroutines
public :: nmlfile                      ! functions
```

3.8 Procedure Interface

3.8.1 Initialize module and acquire NAMELIST

`nmlfile_mod` モジュールの初期化ルーチン。引数 `nml` に渡された文字列を NAMELIST ファイル名として格納し、そのファイルを `nmlfile_open` で `open`、`nmlfile_close` で `close` する。

メインプログラムにおいて、他のモジュールの初期化ルーチンよりも先に呼ばれる事が想定されており、その際に引数 `nml` に渡された NAMELIST ファイル名を以降 (正確には `nmlfile_end` が呼ばれるまで)、他のモジュールでも利用する事となる。

もしも、メインプログラムにおいて呼ばなかった場合や引数無しで呼ばれた場合、または引数に空白を与えた場合は、デフォルトの NAMELIST ファイル *nmlfile.nml* が読み込まれる。

```
subroutine nmlfile_init(nml)
```

Dependency

```
use type_mod, only: STRING, TOKEN, INTKIND
use dc_trace, only: BeginSub, EndSub, DbgMessage
use dc_message,only: MessageNotify
```

Input

```
character(*), intent(in), optional :: nml ! NAMELIST file name
```

3.8.2 Open NAMELIST file, and Return Device Number

nmlfile.mod に格納している NAMELIST ファイルを「適当な」装置番号で Open し、その装置番号を *nmlunit* として返し、*readable* に *.true.* を返す。もしも Open が失敗した、またはファイルが読み取り不可能な場合には *nmlunit* に *-1* を、*readable* に *.false.* を返す。

なお、以前にこのプログラムにて NAMELIST ファイルが Open されている場合、既に Open されている装置番号を自動的に閉じる。

```
subroutine nmlfile_open(nmlunit, readable)
```

Dependency

```
use type_mod, only: STRING, TOKEN, INTKIND
use dc_trace, only: BeginSub, EndSub, DbgMessage
use dc_string, only: CPrintf
use dc_message,only: MessageNotify
```

Output

```
integer(INTKIND), intent(out):: nmlunit ! Device Number for nml file
logical , intent(out):: readable ! Readable Flag
```

3.8.3 Close NAMELIST file

nmlfile_open で開いたファイルを Close する。*nmlfile_open* が呼ばれていない場合には何もせずに終了する。

```
subroutine nmlfile_close()
```

Dependency

```
use type_mod, only: STRING, TOKEN, INTKIND
use dc_trace, only: BeginSub, EndSub, DbgMessage
use dc_string, only: CPrintf
use dc_message,only: MessageNotify
```

3.8.4 Return NAMELIST file name

```
character(TOKEN) function nmlfile() result(result)
```

Dependency

```
use type_mod, only: STRING, TOKEN, INTKIND
use dc_trace, only: BeginSub, EndSub, DbgMessage
use dc_string, only: CPrintf
use dc_message,only: MessageNotify
```

3.8.5 Terminate module

`nmlfile.init` で設定された値を破棄し、読み込む NAMELIST ファイルの名前を `nmlfile.nml` に戻す。

```
subroutine nmlfile_end()
```

Dependency

```
use type_mod, only: STRING
use dc_trace, only: BeginSub, EndSub, DbgMessage
```

4 Module constants_mod

- Developers: Morikawa Yasuhiro
- Version: \$Id: constants.f90,v 1.10 2005/01/20 11:03:46 morikawa Exp \$
- Tag Name: \$Name: \$
- Change History:

4.1 Overview

This module set physical constants. Default value is standard condition on the earth. These values could be changed by NAMELIST.

物理定数を設定するためのモジュールである。デフォルトでは地球上の標準状態の値を持っているが、NAMELIST により変更することも可能である。

4.2 Error Handling

4.3 Known Bugs

4.4 Note

4.5 Future Plans

4.6 Dependency

```
use type_mod, only : REKIND, DBKIND, INTKIND, TOKEN, STRING
```

4.7 Public Interface

```
private
public :: constants_init, constants_end ! subroutines
public :: & ! variables
    & PI, R0, Omega, Grav, Cp, RAir, EL, CpVap, RVap, &
    & DH20, EpsV, ES0, StB, FKarm, EpsVT, SecPerDay, &
    & EFoldTime, TempAve, TimeFilter, VisOrder, &
    & TimeFilterStepInt
```

4.8 Public Data

```
real(DBKIND), save :: & ! follow data is default values.
    &
    & PI          = 3.141592653589793 , & ! 円周率
    & R0          = 6.371d6           , & ! 球の半径
    & Omega       = 7.292d-5         , & ! 回転角速度
    & Grav        = 9.8d0            , & ! 重力加速度
    &
    & Cp          = 1004.6d0         , & ! 大気定圧比熱
    & RAir        = 287.04d0        , & ! 大気気体定数
    & EL          = 2.5d6           , & ! 水の凝結の潜熱
```



```

& CpVap      = 1810.0d0      , & ! 水蒸気定圧比熱
& RVap       = 461.0d0      , & ! 水蒸気気体定数
& DH2O       = 1000.0d0     , & ! 水の密度
& EpsV       = 0.6226464208 , & ! 水と大気の分子量比
& ES0        = 611.0d0     , & ! 0 飽和蒸気圧: P a
& StB        = 5.67d-8     , & ! Stefan-Boltzman
& FKarm      = 0.4d0       , & ! Karman 定数
& EpsVT      = 0.6060479376 , & ! 1/ -1
&
& SecPerDay  = 86400.0d0   , & ! 1日あたりの秒数
&
& EFoldTime  = 8640.0d0    , & ! 最大波数に対する e-folding time
& TempAve   = 300.0       , & ! 平均温度
& TimeFilter= 0.05        ! 時間フィルター係数

integer(INTKIND), save :: &
  & VisOrder      = 4      , & ! 超粘性の次数
  & TimeFilterStepInt = 1  ! 時間フィルターするステップ間隔

```

4.9 Procedure Interface

4.9.1 Initialize module and acquire NAMELIST

モジュールを初期化し、NAMELIST から値を取得する。NAMELIST から値が取得できないものに関しては上記のデフォルト値が用いられる。

NAMELIST ファイルは、メインプログラムにて nmlfile_mod の nmlfile_init で指定されることが想定されているが、もしもこの初期化ルーチンより以前に指定されていなければ、nmlfile_init のデフォルトで指定される NAMELIST ファイルを読む。

```
subroutine constants_init
```

Dependency

```

use type_mod      , only : REKIND, DBKIND, INTKIND, TOKEN, STRING
use nmlfile_mod  , only : nmlfile_init, nmlfile_open, nmlfile_close
use dc_trace     , only : BeginSub, EndSub, DbgMessage
use dc_message   , only : MessageNotify

```

NAMELIST

```

namelist /constants_nml/ &
&
& PI                , & ! 円周率
& R0                , & ! 球の半径
& Omega            , & ! 回転角速度
& Grav             , & ! 重力加速度
&
& Cp               , & ! 大気定圧比熱
& RAir            , & ! 大気気体定数
& EL              , & ! 水の凝結の潜熱
& CpVap          , & ! 水蒸気定圧比熱
& RVap           , & ! 水蒸気気体定数
& DH20           , & ! 水の密度
& EpsV           , & ! 水と大気の分子量比
& ES0            , & ! 0 飽和蒸気圧 : P a
& StB            , & ! Stefan-Boltzman
& FKarm          , & ! Karman 定数
& EpsVT         , & ! 1/ -1
&
& SecPerDay      , & ! 1日あたりの秒数
&
& EFoldTime      , & ! 最大波数に対する e-folding time
& TempAve       , & ! 平均温度
& TimeFilter     , & ! 時間フィルター係数
& TimeFilterStepInt , & ! 時間フィルターするステップ間隔
& VisOrder       ! 超粘性の次数

```

4.9.2 Terminate module

NAMelist から読み込んだ値を破棄し、各定数をデフォルトの値に戻す。

```
subroutine constants_end
```

Dependency

```
use dc_trace,    only : BeginSub, EndSub, DbgMessage
```

5 Module time_mod

- Developers: Morikawa Yasuhiro

- Version: \$Id: time.f90,v 1.10 2005/01/19 08:52:41 morikawa Exp \$
- Tag Name: \$Name: \$
- Change History:

5.1 Overview

This module set "Time" and "Step".
時刻や「計算ステップ」に関して設定するためのモジュールである。

5.2 Error Handling

5.3 Known Bugs

5.4 Note

5.5 Future Plans

5.6 Dependency

```
use type_mod,    only : REKIND, DBKIND, INTKIND, TOKEN, STRING
```

5.7 Public Interface

```
private
public :: time_init, time_progress, time_end, & ! subroutines
        &   InitTime, DelTime,                & ! variables
        &   StepInterval, OutputStep,        & ! variables
        &   CurrentTime, CurrentLoop,        & ! variables
        &   tvar, ttype, tname, tunit        ! variables
```

5.8 Public Data

データ出力の際のステップ間隔や出力回数などに関する情報。time_init で設定。

```
real(DBKIND)    , save :: &
        & InitTime = 0.0    , & ! 積分開始時刻
        & DelTime  = 300    ! 時間ステップ

integer(INTKIND) , save :: &
```

```

& StepInterval = 1 , & ! 出力ステップ間隔
& OutputStep   = 1      ! 出力回数

character(STRING), save :: &
  & tvar = 'time'      , & ! 時刻変数名
  & ttype = 'float'    , & ! 時刻変数の型
  & tname = 'time'    , & ! 時刻の名称
  & tunit = 'seconds' ! 時刻の単位

```

現在の時刻やループ回数を保持してある変数。time_init で初期設定され、time_progress で更新される。

```

real(DBKIND)      , save :: CurrentTime = 0.0 ! 現在時刻
integer(INTKIND) , save :: CurrentLoop = 1   ! 現在のループ回数

```

5.9 Procedure Interface

5.9.1 Initialize module and acquire NAMELIST

モジュールを初期化し、NAMELIST から値を取得する。NAMELIST から値が取得できないものに関しては上記のデフォルト値が用いられる。

NAMELIST ファイルは、メインプログラムにて nmlfile_mod の nmlfile_init で指定されることが想定されているが、もしもこの初期化ルーチンより以前に指定されていなければ、nmlfile_init のデフォルトで指定される NAMELIST ファイルを読む。

```
subroutine time_init
```

Dependency

```

use type_mod,      only : REKIND, DBKIND, INTKIND, TOKEN, STRING
use nmlfile_mod,  only : nmlfile_init, nmlfile_open, nmlfile_close
use dc_trace,     only : BeginSub, EndSub, DbgMessage
use dc_message,   only : MessageNotify

```

NAMELIST

InitTime に与えられた数値に DelTime を加えた値が、CurrentTime に代入される。

```

namelist /time_nml/ &
  &
  & InitTime      , & ! 積分開始時刻

```

```

& DelTime      , &    ! 時間ステップ
&
& StepInterval , &    ! 出力回数
& OutputStep   , &    ! 出力ステップ間隔
&
& tvar         , &    ! 時刻変数名
& ttype        , &    ! 時刻変数の型
& tname        , &    ! 時刻の名称
& tunit        ,      ! 時刻の単位

```

5.9.2 Progress Time and Number of Loop

CurrentTime を time_init で指定された DelTime 分だけ進め、Current-Loop を 1 増やす。今の所、時間を指定したり巻戻したりできない。

```
subroutine time_progress
```

Dependency

```

use type_mod,    only : STRING
use dc_trace,   only : BeginSub, EndSub, DbgMessage

```

5.9.3 Terminate module

NAMELIST から読み込んだ値を破棄し、各定数をデフォルトの値に戻す。

```
subroutine time_end
```

Dependency

```

use type_mod,    only : STRING
use dc_trace,   only : BeginSub, EndSub, DbgMessage

```

6 Module grid_3d_mod

- Developers: Morikawa Yasuhiro
- Version: \$Id: grid_3d.f90,v 1.7 2005/01/19 08:52:30 morikawa Exp \$
- Tag Name: \$Name: \$
- Change History:

6.1 Overview

This module set 3 Dimensional Grid Information.
3次元の格子点情報の設定を行うためのモジュールである。

6.2 Error Handling

6.3 Known Bugs

6.4 Note

6.5 Future Plans

もしかすると、「3次元」という枠組ではなく、それぞれの次元に分けたモジュールにすべきかも知れない。

6.6 Dependency

```
use type_mod, only : INTKIND, STRING
```

6.7 Public Interface

```
private
public :: grid_3d_init, grid_3d_end ! subroutines
public :: im, jm, km             ! variables
```

6.8 Public Data

grid_3d_init において値が設定される。以下はデフォルト値。

```
integer(INTKIND), save :: im = 64 , & ! 格子点の設定 (X)
&                        jm = 32 , & ! 格子点の設定 (Y)
&                        km = 12   ! 格子点の設定 (Z)
```

6.9 Procedure Interface

6.9.1 Initialize module and acquire NAMELIST

NAMELIST から x 軸、y 軸、z 軸の格子点数を取得する。取得できない場合、またはゼロ以下の値が与えられた場合、上記のデフォルト値が強制的に設定される。

NAMELIST ファイルは、メインプログラムにて `nmlfile_mod` の `nmlfile_init` で指定されることが想定されているが、もしもこの初期化ルーチンより以前に指定されていなければ、`nmlfile_init` のデフォルトで指定される NAMELIST ファイルを読む。

```
subroutine grid_3d_init
```

Dependency

```
use type_mod,    only: INTKIND
use nmlfile_mod,only: nmlfile_init, nmlfile_open, nmlfile_close
use dc_trace,    only: DbgMessage, BeginSub, EndSub
use dc_message,  only: MessageNotify
```

NAMELIST

```
namelist /grid_3d_nml/ &
    & im , &      ! 東西格子点数
    & jm , &      ! 南北格子点数
    & km          ! 鉛直格子点数
```

6.9.2 Terminate module

NAMELIST から読み込んだ値を破棄し、各定数をデフォルトの値に戻す。

```
subroutine grid_3d_end
```

Dependency

```
use dc_trace,    only: DbgMessage, BeginSub, EndSub
```

7 Module `grid_wavenumber_mod`

- Developers: Morikawa Yasuhiro
- Version: \$Id: grid_wavenumber.f90,v 1.7 2005/01/19 08:52:30 morikawa Exp \$
- Tag Name: \$Name: \$
- Change History:

7.1 Overview

This module set maximum wavenumber of harmonic function in spectral method.

スペクトル法を用いる際の球面調和関数の最大波数の設定を行う。

7.2 Error Handling

7.3 Known Bugs

7.4 Note

7.5 Future Plans

7.6 Dependency

```
use type_mod, only : INTKIND, STRING
```

7.7 Public Interface

```
private
public :: grid_wavenumber_init, grid_wavenumber_end ! subroutine
public :: nm                                     ! variables
```

7.8 Public Data

grid_wavenumber_init において値が設定される。以下はデフォルト値。

```
integer(INTKIND), save :: nm = 21      ! 波数の設定
```

7.9 Procedure Interface

7.9.1 Initialize module and acquire NAMELIST

NAMELIST から最大波数を取得する。取得できない場合、またはゼロ以下の値が与えられた場合、上記のデフォルト値が強制的に設定される。

NAMELIST ファイルは、メインプログラムにて nmlfile_mod の nmlfile_init で指定されることが想定されているが、もしもこの初期化ルーチンより以前に指定されていなければ、nmlfile_init のデフォルトで指定される NAMELIST ファイルを読む。

```
subroutine grid_wavenumber_init
```


Dependency

```
use type_mod,    only: INTKIND
use nmlfile_mod,only: nmlfile_init, nmlfile_open, nmlfile_close
use dc_trace,    only: DbgMessage, BeginSub, EndSub
use dc_message, only: MessageNotify
```

NAMELIST

```
namelist /grid_wavenumber_nml/ nm ! 最大波数
```

7.9.2 Terminate module

NAMELIST から読み込んだ値を破棄し、各定数をデフォルトの値に戻す。

```
subroutine grid_wavenumber_end
```

Dependency

```
use dc_trace,    only: DbgMessage, BeginSub, EndSub
```

8 Module spml_mod

- Developers: Morikawa Yasuhiro
- Version: \$Id: spml.f90,v 1.6 2005/01/20 11:02:09 morikawa Exp \$
- Tag Name: \$Name: \$
- Change History:

8.1 Overview

This module is wrapper of wa_module in SPMODEL.

SPMODEL ライブラリの wa_module のラッパーモジュールである。他のモジュールから SPMODEL のサブルーチンおよび関数を呼ぶ場合には、原則的にこのモジュールを介して呼び出す。

8.2 Error Handling

8.3 Known Bugs

8.4 Note

`wa_module` の全ての関数をそのまま利用できるよう、`private` をつけていない。実際にこのモジュールで行なっているのは、初期化を複数回行なわなないようにしているだけである。

8.5 Future Plans

8.6 Dependency

このモジュール内部で用いるだけでなく、力学コアなどに `wa_module` の関数群を提供するという意味で、敢えて `only` 属性をつけていない。(本来は必要な全ての関数を指定すべきであろうが)。

```
use wa_module
use w_module,    only : rn
use type_mod,   only : STRING
```

8.7 Public Interface

```
public :: spml_init, spml_end ! subroutines
```

その他、`wa_module` で提供される関数群。

8.8 Procedure Interface

8.8.1 Initialize `wa_module`

`wa_module` の初期化関数 `wa_initial` を呼ぶ。

```
subroutine spml_init
```

Dependency

```
use type_mod           , only: INTKIND, STRING
use grid_3d_mod        , only: im, jm, km, grid_3d_init
use grid_wavenumber_mod, only: nm, grid_wavenumber_init
use dc_trace,         only: DbgMessage, BeginSub, EndSub
use dc_error,         only: StoreError, USR_ECHAR
use wa_module,        only: wa_initial
```

8.8.2 Terminate module

現状では `wa_Initial` を解除できないので、何もしない。`wa_Initial` による初期化を解除できるのなら、`spml_init` による初期化フラグをオフにすべきである。

```
subroutine spml_end
```

Dependency

```
use type_mod,    only: INTKIND, STRING
use dc_trace,    only: DbgMessage, BeginSub, EndSub
```

9 Module `axis_type_mod`

- Developers: Morikawa Yasuhiro
- Version: \$Id: axis_type.f90,v 1.8 2005/01/19 08:52:24 morikawa Exp \$
- Tag Name: \$Name: \$
- Change History:

9.1 Overview

This module provide derived types including all information about Axes data, and utility to treat the derived type variables.

座標軸に関する全ての情報を包括する構造体、およびその構造体変数を扱うためのユーティリティを提供する。

9.2 Error Handling

9.3 Known Bugs

9.4 Note

9.5 Future Plans

9.6 Dependency

```
use type_mod,    only: REKIND, DBKIND, INTKIND, TOKEN, STRING
use gt4_history, only: GT_HISTORY_AXIS, GT_HISTORY_ATTR
```

9.7 Public Interface

```
private
public :: AXISINFO                ! derived types
public :: axis_type_copy, axis_attrs_copy ! subroutines
public :: axis_attrs_init         ! subroutines
```

9.8 Generic Procedure

```
interface axis_type_copy
  module procedure axis_type_copy0, axis_type_copy1
end interface

interface axis_attrs_init
  module procedure axis_attrs_init0, axis_attrs_init1
end interface
```

9.9 Derived Types

次元に関する情報、およびデータを格納

```
type AXISINFO
  type(GT_HISTORY_AXIS)          :: axisinfo ! gt4 変数情報
  type(GT_HISTORY_ATTR), allocatable:: attrs(:) ! 属性情報群
  real(DBKIND)                  , allocatable:: a_Dim(:) ! 次元データ
  logical                        :: stored = .false. ! 格納完了フラグ
end type AXISINFO
```

9.10 Procedure Interface

9.10.1 Copy AXISINFO data (for 0 dimensional data)

```
subroutine axis_type_copy0(from, to)
```

Dependency

```
use type_mod, only: STRING
use dc_trace, only: BeginSub, EndSub
```

Input

```
type(AXISINFO), intent(in) :: from
```

Output

```
type(AXISINFO), intent(out) :: to
```

9.10.2 Copy AXISINFO data (for 1 dimensional data)

```
subroutine axis_type_copy1(from, to)
```

Dependency

```
use type_mod, only: STRING, INTKIND
use dc_trace, only: BeginSub, EndSub
```

Input

```
type(AXISINFO), intent(in) :: from(:)
```

Output

```
type(AXISINFO), intent(out) :: to(:)
```

9.10.3 Copy AXISATTR (in gt4f90io) data (for 1 dimensional data)

```
subroutine axis_attrs_copy(from, to)
```

Dependency

```
use type_mod, only: STRING, INTKIND
use gt4_history, only: GT_HISTORY_ATTR
use dc_trace, only: BeginSub, EndSub, DbgMessage
```

Input

```
type(GT_HISTORY_ATTR), intent(in) :: from(:)
```

Output

```
type(GT_HISTORY_ATTR), intent(out) :: to(:)
```

9.10.4 Initialize AXISATTR (in gt4f90io) data (for 0 dimensional data)

```
subroutine axis_attrs_init0(attrs)
```

Dependency

```
use type_mod, only: STRING, INTKIND
use gt4_history, only: GT_HISTORY_ATTR
use dc_trace, only: BeginSub, EndSub, DbgMessage
```

In/Out

```
type(GT_HISTORY_ATTR), intent(inout):: attrs
```

9.10.5 Initialize AXISATTR (in gt4f90io) data (for 1 dimensional data)

```
subroutine axis_attrs_init1(attrs)
```

Dependency

```
use type_mod, only: STRING, INTKIND
use gt4_history, only: GT_HISTORY_ATTR
use dc_trace, only: BeginSub, EndSub, DbgMessage
```

In/Out

```
type(GT_HISTORY_ATTR), intent(inout):: attrs(:)
```

10 Module axis_x_mod

- Developers: Morikawa Yasuhiro
- Version: \$Id: axis_x.f90,v 1.8 2005/01/19 08:52:24 morikawa Exp \$
- Tag Name: \$Name: \$
- Change History:

10.1 Overview

This module set axis X or axis Longitude.
X 軸または経度軸を設定する。

10.2 Error Handling

10.3 Known Bugs

netCDF データから X 軸を入力する `axis_x_netcdf` にて、元データが `radians` でも `degrees` でも、そのまま入力されるようになっている。本来は元データの `units` から判定すべき。

10.4 Note

10.5 Future Plans

10.6 Dependency

```
use type_mod,      only : INTKIND, STRING
use axis_type_mod, only : AXISINFO
```

10.7 Public Interface

```
private
public :: axis_x_init, axis_x_weight, axis_x_spectral ! subroutines
public :: axis_x_manual, axis_x_netcdf, axis_x_end    ! subroutines
```

10.8 Procedure Interface

10.8.1 Initialize module and acquire NAMELIST

モジュールを初期化し、NAMELIST から値を取得する。NAMELIST から値が取得できないものに関しては上記のデフォルト値が用いられる。

NAMELIST ファイルは、メインプログラムにて `nmlfile_mod` の `nmlfile_init` で指定されることが想定されているが、もしもこの初期化ルーチンより以前に指定されていなければ、`nmlfile_init` のデフォルトで指定される NAMELIST ファイルを読む。

```
subroutine axis_x_init
```

Dependency

```
use type_mod,  only: STRING, TOKEN, INTKIND, REKIND, DBKIND, NMLARRAY
use nmlfile_mod,only: nmlfile_init, nmlfile_open, nmlfile_close
use grid_3d_mod,only: im, grid_3d_init
use spml_mod,   only: spml_init
```

```

use axis_type_mod, only : axis_attrs_copy, axis_attrs_init
use gt4_history, only: GT_HISTORY_ATTR
use dc_types,  only: GT_TOKEN => TOKEN, GT_STRING => STRING
use dc_url,    only: GT_ATMARK, GT_QUESTION
use dc_trace,  only: DbgMessage, BeginSub, EndSub
use dc_message,only: MessageNotify

```

NAMELIST

X 軸の次元変数に関する情報を与える。値を与えないものに関しては以下のデフォルトの値が用いられる。

変数 `decision` には X 軸のデータをどのように与えるかを指定する。

- 'manual'
 - Data 配列に格納したデータをそのまま X 軸として与える。
- 'spectral'
 - スペクトル法を用いる事を想定し、`length` に応じて自動的に X 軸のデータが決まる。
- `gtool4` 変数 (例えば 'foo.nc@lon' など)
 - 該当する変数から X 軸のデータを取得する。

変数 `length` には、`grid3d_mod` の公開要素 `im` と同じ値を与えなければならない。

```

character(TOKEN)  :: name      = 'lon'           ! 次元変数名
integer(INTKIND)  :: length    = 64             ! 次元長 (配列サイズ)
character(String) :: longname  = 'Longitude'    ! 次元変数の記述的名称
character(String) :: units     = 'degrees_east' ! 次元変数の単位
character(TOKEN)  :: xtype     = 'float'        ! 次元変数の型
character(String) :: decision  = 'spectral'     ! 次元データの取得方法
real(REKIND)      :: Data(NMLARRAY) = 0.0      ! 次元データ入力用

```

```

namelist /axis_x_nml/ &
  & name      , & ! 次元変数名
  & length    , & ! 次元長 (配列サイズ)

```



```

& longname      , & ! 次元変数の記述的名称
& units         , & ! 次元変数の単位
& xtype        , & ! 次元変数の型
& decision     , & ! 次元データの取得方法
& Data         , & ! 次元データ

```

X 軸の次元変数の属性に関する情報を与える。NAMELIST に複数の `axis_x_attr_nml` を用意しておく事で複数の情報を与える事が可能である。与えない場合には属性情報は付加されない。

`attrtype` には与える属性値の種類を設定する。<URL:http://www.gfd-dennou.org/arch/gtool4/gt4f90io-current/doc/gt_history.htm#derived_gthistoryattr> を参照せよ。なお、`arraysize` に 1 以上の値を設定すると、配列データが優先されて属性値に設定される。

```

character(GT_TOKEN) :: attrname = '' ! 属性名
character(GT_TOKEN) :: attrtype = '' ! 属性値の型
character(GT_STRING) :: cvalue = '' ! 属性の値 (文字)
integer(INTKIND)    :: ivalue = 0 ! 属性の値 (整数)
real(REKIND)        :: rvalue = 0.0 ! 属性の値 (単精度実数)
real(DBKIND)        :: dvalue = 0.0d0 ! 属性の値 (倍精度実数)
logical             :: lvalue = .false.! 属性の値 (論理)
integer(INTKIND)    :: arraysize= 0 ! 配列のサイズ
integer(INTKIND) :: iarray(NMLARRAY) = 0 ! 属性の値 (整数)
real(REKIND)      :: rarray(NMLARRAY) = 0.0 ! 属性の値 (単精度実数)
real(DBKIND)      :: darray(NMLARRAY) = 0.0d0! 属性の値 (倍精度実数)

```

```

namelist /axis_x_attr_nml/ &
  & attrname      , & ! 属性名
  & attrtype     , & ! 属性値の型
  & cvalue       , & ! 属性の値 (文字)
  & ivalue      , & ! 属性の値 (整数)
  & rvalue      , & ! 属性の値 (単精度実数)
  & dvalue      , & ! 属性の値 (倍精度実数)
  & lvalue      , & ! 属性の値 (論理)
  & arraysize   , & ! 配列のサイズ
  & iarray      , & ! 属性の値 (整数)
  & rarray      , & ! 属性の値 (単精度実数)
  & darray      , & ! 属性の値 (倍精度実数)

```

10.8.2 Return Weight of axis X

重みデータとその付加情報を返す。axis_x_init の NAMELIST axis_x_nml の decision 変数で 'spectral' 以外が与えられた場合は値を代入しないで返す。

また、X 軸の次元変数に重みデータに関する付加情報を加える。

```
subroutine axis_x_weight(Dim_Weight)
```

Dependency

```
use constants_mod, only: constants_init, pi
use axis_type_mod, only: axis_type_copy, axis_attrs_copy, axis_attrs_init
use spml_mod,    only: wa_module_x_Lon_Weight => x_Lon_Weight
use grid_3d_mod,only: im
use gt4_history,only: GT_HISTORY_ATTR
use dc_trace,   only: DbgMessage, BeginSub, EndSub
```

Output

```
type(AXISINFO), intent(out) :: Dim_Weight ! 次元情報を包括する変数
```

10.8.3 Return Data of axis X for Spectral Method.

スペクトル法を用いる場合を想定した X 軸のデータを返す。

axis_x_init の NAMELIST axis_x_nml の units に 'radian' または 'rad.' を与える場合には単位がラジアンでデータが返される。それ以外では度数でデータが返る。

axis_x_init の NAMELIST axis_x_nml の decision 変数で 'spectral' 以外が与えられた場合は値を代入しないで返す。

```
subroutine axis_x_spectral(Dim)
```

Dependency

```
use axis_type_mod, only: axis_type_copy
use constants_mod, only: constants_init, pi
use spml_mod,    only: wa_module_x_Lon => x_Lon
use dc_string,   only: toChar, StrHead, LChar
use dc_trace,   only: DbgMessage, BeginSub, EndSub
```

In/Out

```
type(AXISINFO), intent(inout) :: Dim      ! 次元情報を包括する  
変数
```

10.8.4 Return Data of axis X from NAMELIST

NAMELIST から代入されたデータを X 軸データとして返す。

axis_x_init の NAMELIST axis_x_nml の decision 変数で 'manual' 以外が与えられた場合は値を代入しないで返す。

```
subroutine axis_x_manual(Dim)
```

Dependency

```
use axis_type_mod, only: axis_type_copy  
use spml_mod,    only: wa_module_x_Lon => x_Lon  
use dc_trace,   only: DbgMessage, BeginSub, EndSub
```

In/Out

```
type(AXISINFO), intent(inout) :: Dim      ! 次元情報を包括する  
変数
```

10.8.5 Return Data of axis X from netCDF file

netCDF データから取得したデータを X 軸のデータとして返す。

現在、取得先のデータの単位に関わらず、そのままデータが入力される。

1.3 参照。

axis_x_init の NAMELIST axis_x_nml の decision 変数で gtool4 変数以外が与えられた場合は値を代入しないで返す。

```
subroutine axis_x_netcdf(Dim)
```

Dependency

```
use axis_type_mod, only: axis_type_copy  
use gt4_history, only: HistoryGet  
use dc_url ,    only: UrlSplit  
use dc_trace,  only: DbgMessage, BeginSub, EndSub
```

In/Out

`type(AXISINFO), intent(inout) :: Dim` ! 次元情報を包括する
変数

10.8.6 Terminate module

`axis_x_init` で設定された値を破棄し、デフォルトに戻す。

```
subroutine axis_x_end()
```

Dependency

```
use dc_trace, only: DbgMessage, BeginSub, EndSub
```

11 Module axis_y_mod

- Developers: Morikawa Yasuhiro
- Version: \$Id: axis_y.f90,v 1.9 2005/01/19 08:52:24 morikawa Exp \$
- Tag Name: \$Name: \$
- Change History:

11.1 Overview

This module set axis Y or axis Latitude.

Y 軸または緯度軸を設定する。

11.2 Error Handling

11.3 Known Bugs

netCDF データから Y 軸を入力する `axis_y_netcdf` にて、元データが radians でも degrees でも、そのまま入力されるようになっている。本来は元データの units から判定すべき。

11.4 Note

11.5 Future Plans

11.6 Dependency

```
use type_mod,      only : INTKIND, STRING
use axis_type_mod, only : AXISINFO
```

11.7 Public Interface

```
private
public :: axis_y_init, axis_y_weight, axis_y_spectral ! subroutines
public :: axis_y_manual, axis_y_netcdf, axis_y_end    ! subroutines
```

11.8 Procedure Interface

11.8.1 Initialize module and acquire NAMELIST

モジュールを初期化し、NAMELIST から値を取得する。NAMELIST から値が取得できないものに関しては上記のデフォルト値が用いられる。

NAMELIST ファイルは、メインプログラムにて nmlfile_mod の nmlfile_init で指定されることが想定されているが、もしもこの初期化ルーチンより以前に指定されていないならば、nmlfile_init のデフォルトで指定される NAMELIST ファイルを読む。

```
subroutine axis_y_init
```

Dependency

```
use type_mod,  only: STRING, TOKEN, INTKIND, REKIND, DBKIND, NMLARRAY
use nmlfile_mod,only: nmlfile_init, nmlfile_open, nmlfile_close
use grid_3d_mod,only: jm, grid_3d_init
use spml_mod,   only: spml_init
use axis_type_mod, only : axis_attrs_copy, axis_attrs_init
use gt4_history, only: GT_HISTORY_ATTR
use dc_types,   only: GT_TOKEN => TOKEN, GT_STRING => STRING
use dc_url,     only: GT_ATMARK, GT_QUESTION
use dc_trace,  only: DbgMessage, BeginSub, EndSub
use dc_message,only: MessageNotify
```

NAMELIST

Y 軸の次元変数に関する情報を与える。値を与えないものに関しては以下のデフォルトの値が用いられる。

変数 `decision` には Y 軸のデータをどのように与えるかを指定する。

- 'manual'
 - Data 配列に格納したデータをそのまま Y 軸として与える。
- 'spectral'
 - スペクトル法を用いる事を想定し、`length` に応じて自動的に Y 軸のデータが決まる。
- `gtool4` 変数 (例えば 'foo.nc@lon' など)
 - 該当する変数から Y 軸のデータを取得する。

変数 `length` には、`grid_3d_mod` の公開要素 `jm` と同じ値を与えなければならない。

```
character(TOKEN)  :: name      = 'lat'           ! 次元変数名
integer(INTKIND)  :: length    = 32             ! 次元長 (配列サイズ)
character(STRING) :: longname  = 'Latitude'     ! 次元変数の記述的名称
character(STRING) :: units     = 'degrees_north' ! 次元変数の単位
character(TOKEN)  :: xtype     = 'float'       ! 次元変数の型
character(STRING) :: decision  = 'spectral'     ! 次元データの取得方法
real(REKIND)      :: Data(NMLARRAY) = 0.0     ! 次元データ入力用
```

```
namelist /axis_y_nml/ &
  & name      , & ! 次元変数名
  & length    , & ! 次元長 (配列サイズ)
  & longname  , & ! 次元変数の記述的名称
  & units     , & ! 次元変数の単位
  & xtype     , & ! 次元変数の型
  & decision  , & ! 次元データの取得方法
  & Data      ! 次元データ
```

Y 軸の次元変数の属性に関する情報を与える。NAMELIST に複数の axis_y_attr_nml を用意しておく事で複数の情報を与える事が可能である。与えない場合には属性情報は付加されない。

attrtype には与える属性値の種類を設定する。<URL:http://www.gfd-dennou.org/arch/gtool4/gt4f90io-current/doc/gt_history.htm#derived_gthistoryattr> を参照せよ。なお、arraysize に 1 以上の値を設定すると、配列データが優先されて属性値に設定される。

```

character(GT_TOKEN)  :: attrname = '' ! 属性名
character(GT_TOKEN)  :: attrtype = '' ! 属性値の型
character(GT_STRING) :: cvalue   = '' ! 属性の値 (文字)
integer(INTKIND)     :: ivalue   = 0   ! 属性の値 (整数)
real(REKIND)         :: rvalue   = 0.0 ! 属性の値 (単精度実数)
real(DBKIND)         :: dvalue   = 0.0d0 ! 属性の値 (倍精度実数)
logical              :: lvalue   = .false.! 属性の値 (論理)
integer(INTKIND)     :: arraysize= 0   ! 配列のサイズ
integer(INTKIND)    :: iarray(NMLARRAY) = 0   ! 属性の値 (整数)
real(REKIND)       :: rarray(NMLARRAY) = 0.0 ! 属性の値 (単精度実数)
real(DBKIND)       :: darray(NMLARRAY) = 0.0d0! 属性の値 (倍精度実数)

```

```

namelist /axis_y_attr_nml/ &
  & attrname      , & ! 属性名
  & attrtype      , & ! 属性値の型
  & cvalue        , & ! 属性の値 (文字)
  & ivalue        , & ! 属性の値 (整数)
  & rvalue        , & ! 属性の値 (単精度実数)
  & dvalue        , & ! 属性の値 (倍精度実数)
  & lvalue        , & ! 属性の値 (論理)
  & arraysize     , & ! 配列のサイズ
  & iarray        , & ! 属性の値 (整数)
  & rarray        , & ! 属性の値 (単精度実数)
  & darray        , & ! 属性の値 (倍精度実数)

```

11.8.2 Return Weight of axis Y

重みデータとその付加情報を返す。axis_y_init の NAMELIST axis_y_nml の decision 変数で 'spectral' 以外が与えられた場合は値を代入しないで返す。

また、Y 軸の次元変数に重みデータに関する付加情報を加える。

```
subroutine axis_y_weight(Dim_Weight)
```

Dependency

```
use constants_mod, only: constants_init, pi
use axis_type_mod, only: axis_type_copy, axis_attrs_copy, axis_attrs_init
use spml_mod,    only: wa_module_y_Lat_Weight => y_Lat_Weight
use grid_3d_mod,only: jm
use gt4_history,only: GT_HISTORY_ATTR
use dc_trace,   only: DbgMessage, BeginSub, EndSub
```

Output

```
type(AXISINFO), intent(out) :: Dim_Weight ! 次元情報を包括する変数
```

11.8.3 Return Data of axis Y for Spectral Method.

スペクトル法を用いる場合を想定した Y 軸のデータを返す。

axis_y_init の NAMELIST axis_y_nml の units に 'radian' または 'rad.' を与える場合には単位がラジアンでデータが返される。それ以外では度数でデータが返る。

axis_y_init の NAMELIST axis_y_nml の decision 変数で 'spectral' 以外が与えられた場合は値を代入しないで返す。

```
subroutine axis_y_spectral(Dim)
```

Dependency

```
use axis_type_mod, only: axis_type_copy
use constants_mod, only: constants_init, pi
use spml_mod,    only: wa_module_y_Lat => y_Lat
use dc_string,  only: toChar, StrHead, LChar
use dc_trace,   only: DbgMessage, BeginSub, EndSub
```

In/Out

```
type(AXISINFO), intent(inout) :: Dim          ! 次元情報を包括する変数
```


11.8.4 Return Data of axis Y from NAMELIST

NAMELIST から代入されたデータを Y 軸データとして返す。

axis_y_init の NAMELIST axis_y_nml の decision 変数で 'manual' 以外が与えられた場合は値を代入しないで返す。

```
subroutine axis_y_manual(Dim)
```

Dependency

```
use axis_type_mod, only: axis_type_copy
use spml_mod,    only: wa_module_y_Lat => y_Lat
use dc_trace,   only: DbgMessage, BeginSub, EndSub
```

In/Out

```
type(AXISINFO), intent(inout) :: Dim      ! 次元情報を包括する
変数
```

11.8.5 Return Data of axis Y from netCDF file

netCDF データから取得したデータを Y 軸のデータとして返す。

現在、取得先のデータの単位に関わらず、そのままデータが入力される。

1.3 参照。

axis_y_init の NAMELIST axis_y_nml の decision 変数で gtool4 変数以外が与えられた場合は値を代入しないで返す。

```
subroutine axis_y_netcdf(Dim)
```

Dependency

```
use axis_type_mod, only: axis_type_copy
use gt4_history,only: HistoryGet
use dc_url ,    only: UrlSplit
use dc_trace,   only: DbgMessage, BeginSub, EndSub
```

In/Out

```
type(AXISINFO), intent(inout) :: Dim      ! 次元情報を包括する
変数
```

11.8.6 Terminate module

axis_y_init で設定された値を破棄し、デフォルトに戻す。

```
subroutine axis_y_end()
```

Dependency

```
use dc_trace, only: DbgMessage, BeginSub, EndSub
```

12 Module axis_z_mod

- Developers: Morikawa Yasuhiro
- Version: \$Id: axis_z.f90,v 1.9 2005/01/19 08:52:24 morikawa Exp \$
- Tag Name: \$Name: \$
- Change History:

12.1 Overview

This module set axis Z or axis Altitude.

Z 軸または高度軸を設定する。

12.2 Error Handling

12.3 Known Bugs

- netCDF データから Z 軸を入力する axis_z_netcdf にて、および axis_z_half_netcdf にて、元データの units によらずデータを入力しているが、本来は判定すべき。(特に圧力座標を扱う場合には必須となるはず)。
- 現在は半整数レベルもこのモジュールで読み込んでいるが、別モジュールにすべきかも知れない。(「バグ情報」と違うが)。

12.4 Note

12.5 Future Plans

12.6 Dependency

```
use type_mod,      only : INTKIND, STRING
use axis_type_mod, only : AXISINFO
```

12.7 Public Interface

```
private
public :: axis_z_init, axis_z_end      ! subroutines
public :: axis_z_manual, axis_z_netcdf ! subroutines
public :: axis_z_sigmahalf_manual     ! subroutines
public :: axis_z_sigmahalf_netcdf     ! subroutines
public :: axis_z_half_manual          ! subroutines
public :: axis_z_half_netcdf          ! subroutines
```

12.8 Procedure Interface

12.8.1 Initialize module and acquire NAMELIST

モジュールを初期化し、NAMELIST から値を取得する。NAMELIST から値が取得できないものに関しては上記のデフォルト値が用いられる。

NAMELIST ファイルは、メインプログラムにて nmlfile_mod の nmlfile_init で指定されることが想定されているが、もしもこの初期化ルーチンより以前に指定されていないならば、nmlfile_init のデフォルトで指定される NAMELIST ファイルを読む。

```
subroutine axis_z_init
```

Dependency

```
use type_mod      ,only: STRING, TOKEN, INTKIND, REKIND, DBKIND, NMLARRAY
use nmlfile_mod,only: nmlfile_init, nmlfile_open, nmlfile_close
use grid_3d_mod,only: km, grid_3d_init
use axis_type_mod, only : axis_attrs_copy, axis_attrs_init
use constants_mod,only: constants_init
use gt4_history,  only: GT_HISTORY_ATTR
use dc_types,    only: GT_TOKEN => TOKEN, GT_STRING => STRING
use dc_url       ,only: GT_ATMARK, GT_QUESTION
use dc_string    ,only: toChar
use dc_trace     ,only: DbgMessage, BeginSub, EndSub
use dc_message  ,only: MessageNotify
```

NAMELIST axis_z_nml

axis_z_nml には、Z 軸の次元変数に関する情報を与える。値を与えないものに関しては以下のデフォルトの値が用いられる。

変数 decision には Z 軸のデータをどのように与えるかを指定する。

- 'manual'
 - Data 配列に格納したデータをそのまま Z 軸として与える。
- 'sigmahalf'
 - Z 軸が整数 レベルであると想定し、axis_z_half_nml で与えられる半整数 レベルの値から、自動的に Z 軸のデータを決める。よって、axis_z_half_nml に有効なデータが与えられない場合にはデータは定まらない。
- gtool4 変数 (例えば 'foo.nc@lon' など)
 - 該当する変数から Z 軸のデータを取得する。
- 上記以外
 - 現在は 'sigmahalf' の場合と同様に設定される

変数 length には、grid_3d_mod の公開要素 km と同じ値を与えなければならない。

```

character(TOKEN)  :: name      = 'sigma'          ! 次元変数名
integer(INTKIND)  :: length    = 12              ! 次元長 (配列サイズ)
character(STRING) :: longname  = 'sigma at full level' ! 次元変数の記述的名称
character(STRING) :: units     = 'sigma_level'    ! 次元変数の単位
character(TOKEN)  :: xtype     = 'float'         ! 次元変数の型
character(STRING) :: decision  = 'sigmahalf'     ! 次元データの取得方法
real(REKIND)      :: Data(NMLARRAY) = 0.0! 次元データ入力用

namelist /axis_z_nml/ &
  & name      , & ! 次元変数名
  & length    , & ! 次元長 (配列サイズ)
  & longname  , & ! 次元変数の記述的名称
  & units     , & ! 次元変数の単位
  & xtype     , & ! 次元変数の型
  & decision  , & ! 次元データの取得方法
  & Data      ! 次元データ

```

NAMELIST axis_z_half_nml

axis_z_half_nml には、Z 軸の半整数レベルの次元変数に関する情報を与える。値を与えないものに関しては以下のデフォルトの値が用いられる。

変数 decision には Z 軸のデータをどのように与えるかを指定する。

- 'manual'
 - Data 配列に格納したデータをそのまま Z 軸半整数レベルとして与える。
- gtool4 変数 (例えば 'foo.nc@lon' など)
 - 該当する変数から Z 軸のデータを取得する。
- 上記以外
 - 現在は 'manual' の場合と同様に設定される

変数 length には、grid_3d_mod の公開要素 km にプラス 1 した値を与なければならない。

```
name          = 'sigmahalf'    ! 次元変数名
length        = 13             ! 次元長 (配列サイズ)
longname      = 'sigma at half level' ! 次元変数の記述的名称
units         = 'sigma_level'  ! 次元変数の単位
xtype         = 'float'        ! 次元変数の型
decision      = 'manual'       ! 次元データの取得方法
Data(1:13)    = (/1, 0.99, 0.97, 0.93, 0.85, 0.75, 0.63, 0.5, &
&                0.36, 0.22, 0.1, 0.05, 0/) ! 次元データ入力
用
```

```
namelist /axis_z_half_nml/ &
& name          , & ! 次元変数名
& length        , & ! 次元長 (配列サイズ)
& longname      , & ! 次元変数の記述的名称
& units         , & ! 次元変数の単位
& xtype         , & ! 次元変数の型
& decision      , & ! 次元データの取得方法
& Data          ! 次元データ
```

NAMELIST axis_z_attr_nml

Z 軸の次元変数の属性に関する情報を与える。NAMELIST に複数の axis_z_attr_nml を用意しておく事で複数の情報を与える事が可能である。与えない場合には属性情報は付加されない。

attrtype には与える属性値の種類を設定する。<URL:http://www.gfd-dennou.org/arch/gtool4/gt4f90io-current/doc/gt_history.htm#derived_gthistoryattr> を参照せよ。なお、arraysize に 1 以上の値を設定すると、配列データが優先されて属性値に設定される。

```
character(GT_TOKEN)  :: attrname = '' ! 属性名
character(GT_TOKEN)  :: attrtype = '' ! 属性値の型
character(GT_STRING) :: cvalue   = '' ! 属性の値 (文字)
integer(INTKIND)     :: ivalue   = 0    ! 属性の値 (整数)
real(REKIND)         :: rvalue   = 0.0  ! 属性の値 (単精度実数)
real(DBKIND)         :: dvalue   = 0.0d0 ! 属性の値 (倍精度実数)
logical              :: lvalue   = .false.! 属性の値 (論理)
integer(INTKIND)     :: arraysize= 0    ! 配列のサイズ
integer(INTKIND)    :: iarray(NMLARRAY) = 0    ! 属性の値 (整数)
real(REKIND)        :: rarray(NMLARRAY) = 0.0  ! 属性の値 (単精度実数)
real(DBKIND)        :: darray(NMLARRAY) = 0.0d0! 属性の値 (倍精度実数)
```

```
namelist /axis_z_attr_nml/ &
  & attrname      , & ! 属性名
  & attrtype     , & ! 属性値の型
  & cvalue       , & ! 属性の値 (文字)
  & ivalue       , & ! 属性の値 (整数)
  & rvalue       , & ! 属性の値 (単精度実数)
  & dvalue       , & ! 属性の値 (倍精度実数)
  & lvalue       , & ! 属性の値 (論理)
  & arraysize    , & ! 配列のサイズ
  & iarray       , & ! 属性の値 (整数)
  & rarray       , & ! 属性の値 (単精度実数)
  & darray       , & ! 属性の値 (倍精度実数)
```

NAMELIST axis_z_half_attr_nml

Z 軸の半整数レベルの次元変数の属性に関する情報を与える。NAMELIST に複数の axis_z_half_attr_nml を用意しておく事で複数の情報を与える事が可能である。与えない場合には属性情報は付加されない。

attrtype には与える属性値の種類を設定する。<URL:http://www.gfd-dennou.org/arch/gtool4/gt4f90io-current/doc/gt_history.htm#derived_gthistoryattr>を参照せよ。なお、arraysize に 1 以上の値を設定すると、配列データが優先されて属性値に設定される。

```
namelist /axis_z_half_attr_nml/ &
  & attrname      , & ! 属性名
  & attrtype      , & ! 属性値の型
  & cvalue        , & ! 属性の値 (文字)
  & ivalue        , & ! 属性の値 (整数)
  & rvalue        , & ! 属性の値 (単精度実数)
  & dvalue        , & ! 属性の値 (倍精度実数)
  & lvalue        , & ! 属性の値 (論理)
  & arraysize     , & ! 配列のサイズ
  & iarray        , & ! 属性の値 (整数)
  & rarray        , & ! 属性の値 (単精度実数)
  & darray        , & ! 属性の値 (倍精度実数)
```

12.8.2 Return Sigma Full Level and Sigma Half Level (from NAMELIST) as Data of axis Z

NAMELIST から取得した半整数 レベルのデータと、その半整数 レベルから生成した整数 レベルデータを返す。

axis_z_init の NAMELIST axis_z_nml の decision 変数で 'sigmahalf' が与えられ、且つ NAMELIST axis_z_half_nml の decision 変数で 'manual' が与えられている場合以外は、値に何も代入せず返す。

```
subroutine axis_z_sigmahalf_manual(Dim, DimHalf)
```

Dependency

```
use axis_type_mod, only: axis_type_copy
use dc_trace,    only: DbgMessage, BeginSub, EndSub
```

In/Out

```
type(AXISINFO), intent(inout) :: Dim, DimHalf ! 次元情報を含む
変数
```

12.8.3 Return Sigma Full Level and Sigma Half Level (from netCDF Data) as Data of axis Z

netCDF データから取得した半整数 レベルのデータと、その半整数 レベルから生成した整数 レベルデータを返す。

axis_z_init の NAMELIST axis_z_nml の decision 変数で 'sigmahalf' が与えられ、且つ NAMELIST axis_z_half_nml の decision 変数で gtool4 変数が与えられている場合以外は、値に何も代入せず返す。

```
subroutine axis_z_sigmahalf_netcdf(Dim, DimHalf)
```

Dependency

```
use axis_type_mod, only: axis_type_copy
use gt4_history, only: HistoryGet
use dc_url , only: UrlSplit
use dc_trace, only: DbgMessage, BeginSub, EndSub
```

In/Out

type(AXISINFO), intent(inout) :: Dim, DimHalf ! 次元情報を包括する変数

12.8.4 Return Data of axis Z from NAMELIST

NAMELIST から代入されたデータを Z 軸データとして返す。

axis_z_init の NAMELIST axis_z_nml の decision 変数で 'manual' 以外が与えられた場合は値を代入しないで返す。

```
subroutine axis_z_manual(Dim)
  use axis_type_mod, only: axis_type_copy
  use dc_trace, only: DbgMessage, BeginSub, EndSub
```

In/Out

type(AXISINFO), intent(inout) :: Dim ! 次元情報を包括する変数

12.8.5 Return Data of axis Z from netCDF file

netCDF データから取得したデータを Z 軸のデータとして返す。

axis_z_init の NAMELIST axis_z_nml の decision 変数で gtool4 変数以外が与えられた場合は値を代入しないで返す。


```
subroutine axis_z_netcdf(Dim)
```

Dependency

```
use type_mod,      only: STRING
use axis_type_mod, only: axis_type_copy
use gt4_history, only: HistoryGet
use dc_url ,       only: UrlSplit
use dc_trace,     only: DbgMessage, BeginSub, EndSub
```

In/Out

```
type(AXISINFO), intent(inout) :: Dim      ! 次元情報を包括する
変数
```

12.8.6 Return Half Level Data of axis Z from NAMELIST

NAMELIST から代入された半整数レベルのデータを半整数レベルの Z 軸データとして返す。

axis_z_init の NAMELIST axis_z_half_nml の decision 変数で 'manual' 以外が与えられた場合は値を代入しないで返す。

```
subroutine axis_z_half_manual(DimHalf)
  use axis_type_mod, only: axis_type_copy
  use dc_trace,     only: DbgMessage, BeginSub, EndSub
```

In/Out

```
type(AXISINFO), intent(inout) :: DimHalf  ! 次元情報を包括
する変数
```

12.8.7 Return Half Level Data of axis Z from netCDF file

netCDF データから取得した半整数レベルデータを半整数レベルの Z 軸のデータとして返す。

axis_z_init の NAMELIST axis_z_half_nml の decision 変数で gtool4 変数以外が与えられた場合は値を代入しないで返す。

```
subroutine axis_z_half_netcdf(DimHalf)
```

Dependency

```
use type_mod,      only: STRING
use axis_type_mod, only: axis_type_copy
use gt4_history, only: HistoryGet
use dc_url ,      only: UrlSplit
use dc_trace,    only: DbgMessage, BeginSub, EndSub
```

In/Out

```
type(AXISINFO), intent(inout) :: DimHalf      ! 次元情報を包括
する変数
```

12.8.8 Generate Sigma Level from Sigma Half Level.

axis_z_mod の内部サブルーチン。

引数 DimHalf を半整数 レベルとして、その値から整数 レベルを生成し、引数 Dim として返す。

```
subroutine gen_sigma_from_half(DimHalf, Dim)
```

Dependency

```
use type_mod,      only: DBKIND, STRING, REKIND
use constants_mod, only: RAir, Cp
use dc_string,     only: toChar
use dc_trace,     only: DbgMessage, BeginSub, EndSub
```

Input

```
type(AXISINFO), intent(in)   :: DimHalf ! 半整数 レベル
```

In/Out

```
type(AXISINFO), intent(inout):: Dim      ! 整数 レベル
```

12.8.9 Terminate module

axis_z_init で設定された値を破棄し、デフォルトに戻す。

```
subroutine axis_z_end
```

Dependency

```
use dc_trace, only: DbgMessage, BeginSub, EndSub
```

13 Module varinfo_mod

- Developers: Morikawa Yasuhiro
- Version: \$Id: varinfo.f90,v 1.8 2005/01/19 08:52:45 morikawa Exp \$
- Tag Name: \$Name: \$
- Change History:

13.1 Overview

This module store Variable Infomation from NAMELIST and Provide. These Information is expected to be used by io_gt4_out_mod .

NAMELIST から出力する変数に関する情報を取得する。これらの情報は io_gt4_out_mod にて用いられることが想定されている。

13.2 Error Handling

13.3 Known Bugs

13.4 Note

13.5 Future Plans

もしかすると、io_gt4_out_mod モジュール内に格納されるべきかも知れませんが、

13.6 Dependency

```
use type_mod,      only : INTKIND, STRING, TOKEN
use gt4_history,   only : GT_HISTORY_VARINFO, GT_HISTORY_ATTR
```

13.7 Public Interface

```
private
public :: VAR_INFO                                ! derived types
public :: varinfo_init, varinfo_inquire, varinfo_end ! subroutines
```

13.8 Generic Procedure

```
interface varinfo_attrs_init
  module procedure varinfo_attrs_init0, varinfo_attrs_init1
end interface
```

13.9 Derived Types

変数に関する各情報を格納 (データ本体は含まない)

```
type VAR_INFO
  character(STRING)      :: varkey = '' ! 変数キー
  character(STRING)      :: file   = '' ! 出力するファイル
  integer(INTKIND)       :: StepInterval = 0 ! 出力ステップ間隔

  integer(INTKIND)       :: OutputStep = 0 ! 出力回数
  type(GT_HISTORY_VARINFO):: varinfo   ! gt4 変数情報
  type(GT_HISTORY_ATTR), allocatable &
    &                          :: attrs(:) ! 属性情報群
end type VAR_INFO
```

13.10 Procedure Interface

13.10.1 Initialize module and acquire NAMELIST

モジュールを初期化し、NAMELIST から値を取得する。NAMELIST から値が取得できないものに関しては上記のデフォルト値が用いられる。

NAMELIST ファイルは、メインプログラムにて nmlfile_mod の nmlfile_init で指定されることが想定されているが、もしもこの初期化ルーチンより以前に指定されていなければ、nmlfile_init のデフォルトで指定される NAMELIST ファイルを読む。

```
subroutine varinfo_init
```

Dependency

```
use type_mod, only: STRING, TOKEN, INTKIND, REKIND, DBKIND, &
  &                NMLARRAY
use nmlfile_mod, only: nmlfile_init, nmlfile_open, nmlfile_close
use dc_types, only: GT_TOKEN => TOKEN, GT_STRING => STRING
use dc_trace, only: DbgMessage, BeginSub, EndSub
use dc_message, only: MessageNotify
```

NAMELIST varinfo_nml

変数に関する基本情報を設定するための NAMELIST で、複数の varinfo_nml を用意する事で、複数の変数の設定が可能である。変数 varkey はモデル毎に内部で設定される物理量のマーカで、具体的には io_gt4_out_mod モジュールの io_gt4_out_SetVars サブルーチンでマーカを設定し、io_gt4_out_Put サブルーチンでデータを file に指定されたファイルに出力する。もしも file を与えない、または空文字を与えた場合には、io_gt4_out_nml で与えた default_output 変数で指定されたファイルに出力する。

varname, dimnum, dimnames, longname, units, xtype は出力される変数に付加される情報である。

StepInterval, OutputStep を与えない、またはゼロ以下の値を与えた場合には time_mod の time_nml で与えた StepInterval, OutputStep が用いられる。

```
character (STRING)    :: varkey   = '' ! 変数キー
character (STRING)    :: file     = '' ! 出力するファイル

character (GT_TOKEN)  :: varname  = '' ! 変数名
integer (INTKIND)     :: dimnum   = 0 ! 依存する次元
character (GT_TOKEN)  :: dimnames (NMLARRAY) = '' ! 依存する次元
character (GT_STRING) :: longname = '' ! 変数の記述的名称
character (GT_STRING) :: units    = '' ! 変数の単位
character (GT_TOKEN)  :: xtype    = '' ! 変数の型

integer (INTKIND)     :: StepInterval = 0 ! 出力ステップ間隔
integer (INTKIND)     :: OutputStep   = 0 ! 出力回数

namelist /varinfo_nml/ &
  & varkey      , & ! 変数キー
  & file       , & ! 出力するファイル
  &
  & varname    , & ! 変数名
  & dimnum     , & ! 依存する次元の数
  & dimnames   , & ! 依存する次元変数名
  & longname   , & ! 変数の記述的名称
  & units      , & ! 変数の単位
  & xtype     , & ! 変数の型
  &
  & StepInterval , & ! 出力回数
  & OutputStep   ! 出力ステップ間隔
```

NAMELIST varinfo_attr_nml

変数 varattr の属性情報を与える。NAMELIST に複数の varinfo_attr_nml を用意しておく事で複数の変数に対し、複数の情報を与える事が可能である。与えない場合には属性情報は付加されない。

attrtype には与える属性値の種類を設定する。<URL:http://www.gfd-dennou.org/arch/gtool4/gt4f90io-current/doc/gt_history.htm#derived_gthistoryattr> を参照せよ。なお、arraysize に 1 以上の値を設定すると、配列データが優先されて属性値に設定される。

```
character(GT_STRING) :: varattr = '' ! 属性を付加する変数名
character(GT_TOKEN)  :: attrname = '' ! 属性名
character(GT_TOKEN)  :: attrtype = '' ! 属性値の型
character(GT_STRING) :: cvalue   = '' ! 属性の値 (文字)
integer(INTKIND)     :: ivalue   = 0    ! 属性の値 (整数)
real(REKIND)         :: rvalue   = 0.0  ! 属性の値 (単精度実数)
real(DBKIND)         :: dvalue   = 0.0d0 ! 属性の値 (倍精度実数)
logical              :: lvalue   = .false. ! 属性の値 (論理)
integer(INTKIND)     :: arraysize= 0    ! 配列のサイズ
integer(INTKIND) :: iarray(NMLARRAY) = 0    ! 属性の値 (整数)
real(REKIND)      :: rarray(NMLARRAY) = 0.0  ! 属性の値 (単精度実数)
real(DBKIND)      :: darray(NMLARRAY) = 0.0d0 ! 属性の値 (倍精度実数)
```

```
namelist /varinfo_attr_nml/ &
    & varattr      , & ! 属性を付加する変数名
    & attrname    , & ! 属性名
    & attrtype    , & ! 属性値の型
    & cvalue      , & ! 属性の値 (文字)
    & ivalue      , & ! 属性の値 (整数)
    & rvalue      , & ! 属性の値 (単精度実数)
    & dvalue      , & ! 属性の値 (倍精度実数)
    & lvalue      , & ! 属性の値 (論理)
    & arraysize   , & ! 配列のサイズ
    & iarray      , & ! 属性の値 (整数)
    & rarray      , & ! 属性の値 (単精度実数)
    & darray      , & ! 属性の値 (倍精度実数)
```

13.10.2 Return Variable Information about varkey.

変数キー varkey に対応する変数情報 info を返す。stat にはステータスを返す。正常に変数情報が返れば 0, varkey に対応する情報が無ければ 1, varinfo_init による初期化が行なわれていない場合は -1 を返す。

```
subroutine varinfo_inquire(varkey, info, stat)
```

Dependency

```
use type_mod,    only: STRING, TOKEN, INTKIND, REKIND, DBKIND
use dc_trace,    only: DbgMessage, BeginSub, EndSub
use dc_error,    only: StoreError, USR_ECHAR
```

Input

```
character(*),    intent(in)  :: varkey ! 変数キー
```

Output

```
type(VAR_INFO),  intent(out) :: info   ! VAR_INFO 変数
integer(INTKIND), intent(out) :: stat   ! ステータス
```

13.10.3 Terminate module

varinfo_init で設定された値を破棄し、デフォルトに戻す。

```
subroutine varinfo_end
```

Dependency

```
use dc_trace,    only: DbgMessage, BeginSub, EndSub
```

13.10.4 Copy VAR.INFO array.

varinfo_mod の内部サブルーチン。構造体 VAR.INFO の 1 次元配列のコピーを行なう。

```
subroutine varinfo_copy(from, to)
```

Dependency

```
use type_mod     , only: STRING, INTKIND
use dc_trace     , only: BeginSub, EndSub, DbgMessage
```

Input

```
type(VAR_INFO), intent(in) :: from(:)
```

Output

```
type(VAR_INFO), intent(out) :: to(:)
```

13.10.5 Copy GT_HISTORY_ATTR array.

varinfo_mod の内部サブルーチン。構造体 GT_HISTORY_ATTR の 1 次元配列のコピーを行なう。

```
subroutine varinfo_attrs_copy(from, to)
```

Dependency

```
use type_mod, only: STRING, INTKIND
use gt4_history, only: GT_HISTORY_ATTR
use dc_trace, only: BeginSub, EndSub, DbgMessage
```

Input

```
type(GT_HISTORY_ATTR), intent(in) :: from(:)
```

Output

```
type(GT_HISTORY_ATTR), intent(out) :: to(:)
```

13.10.6 Initialize GT_HISTORY_ATTR variable.

varinfo_mod の内部サブルーチン。構造体 GT_HISTORY_ATTR 変数の初期化を行なう。

```
subroutine varinfo_attrs_init0(attrs)
```

Dependency

```
use type_mod, only: STRING, INTKIND
use gt4_history, only: GT_HISTORY_ATTR
use dc_trace, only: BeginSub, EndSub, DbgMessage
```


In/Out

```
type(GT_HISTORY_ATTR), intent(inout):: attrs
```

13.10.7 Initialize GT_HISTORY_ATTR variable.

varinfo_mod の内部サブルーチン。構造体 GT_HISTORY_ATTR の 1 次元配列の初期化を行なう。

```
subroutine varinfo_attrs_init1(attrs)
```

Dependency

```
use type_mod, only: STRING, INTKIND
use gt4_history, only: GT_HISTORY_ATTR
use dc_trace, only: BeginSub, EndSub, DbgMessage
```

In/Out

```
type(GT_HISTORY_ATTR), intent(inout):: attrs(:)
```

14 Module io_gt4_out_mod

- Developers: Morikawa Yasuhiro
- Version: \$Id: io_gt4_out.f90,v 1.10 2005/01/19 08:52:36 morikawa Exp \$
- Tag Name: \$Name: \$
- Change History:

14.1 Overview

This module output data with gtool4 netCDF conventions.
gtool4 netCDF 規約に基づくデータを出力する。

14.2 Error Handling

14.3 Known Bugs

- 1つのファイルに異なる StepInterval が設定される場合、HistoryCreate には先に設定されたものの StepInterval で時間間隔を決めるため、後

で設定した変数の StepInterval が前のものよりも小さい場合には整合的でなくなる。

– 回避方法

* 異なる StepInterval を設定する場合にはそれらは別個のファイルに分ける。

14.4 Note

14.5 Future Plans

14.6 Dependency

```
use type_mod,      only : REKIND, DBKIND, INTKIND, TOKEN, STRING
use axis_type_mod, only : AXISINFO
use gt4_history,   only : GT_HISTORY
```

14.7 Public Interface

```
private
public :: io_gt4_out_init, io_gt4_out_end      ! subroutines
public :: io_gt4_out_SetDims                  ! subroutines
public :: io_gt4_out_SetVars, io_gt4_out_Put ! subroutines
```

14.8 Generic Procedure

```
interface io_gt4_out_Put
  module procedure io_gt4_out_Put3Real
  module procedure io_gt4_out_Put2Real
  module procedure io_gt4_out_Put0Real
  module procedure io_gt4_out_Put3Double
  module procedure io_gt4_out_Put2Double
  module procedure io_gt4_out_Put0Double
end interface
```

14.9 Derived Types

出力変数データ格納用構造体。 io_gt4_out_SetVars で設定。

```
type IO_GT4_OUT_VARS
```

```

character (STRING), allocatable :: varkeys(:) ! 変数キー
character (STRING)              :: file      ! 出力ファイル
type (GT_HISTORY)               :: gt_history ! GT_HISTORY 変
数
logical                          :: created  ! HistoryCreate 済
かどうか
type (IO_GT4_OUT_VARS), pointer :: next
end type IO_GT4_OUT_VARS

```

14.10 Procedure Interface

14.10.1 Initialize module and acquire NAMELIST

モジュールを初期化し、NAMELIST から値を取得する。NAMELIST から値が取得できないものに関しては上記のデフォルト値が用いられる。

NAMELIST ファイルは、メインプログラムにて nmlfile_mod の nmlfile_init で指定されることが想定されているが、もしもこの初期化ルーチンより以前に指定されていなければ、nmlfile_init のデフォルトで指定される NAMELIST ファイルを読む。

```
subroutine io_gt4_out_init
```

Dependency

```

use type_mod,    only : REKIND, DBKIND, INTKIND, TOKEN, STRING
use nmlfile_mod, only : nmlfile_init, nmlfile_open, nmlfile_close
use time_mod,    only : time_init
use varinfo_mod, only : varinfo_init
use dc_trace,    only : SetDebug, BeginSub, EndSub, DbgMessage
use dc_message,  only : MessageNotify

```

NAMELIST

出力ファイル設定。file に与えたものがデフォルトの出力ファイルとなる。その他の情報は出力する gtool4 netCDF データの大域データとして与えられる。

```

character (STRING) ::
    & file      = 'result.nc'      , & ! 出力ファイル名 (デ
フォルト)
    & title     = 'GCM Test'       , & ! タイトル
    & source    = 'DCPAM'          , & ! モデル名 (作成手段)

```

```
& institution = 'GFD Dennou Club'      ! 実行者名 (作成者)
```

```
namelist /io_gt4_out_nml/ &  
  & file      , & ! 出力ファイル名 (デフォルト)  
  & title     , & ! タイトル  
  & source    , & ! モデル名 (作成手段)  
  & institution ! 実行者名 (作成者)
```

14.10.2 Set Dimension

出力する gtool4 netCDF データの座標情報を設定する。複数回呼ぶ事で複数の座標を設定する。現在の所、設定した座標は出力する全ての netCDF ファイルに出力される。

```
subroutine io_gt4_out_SetDims(axis)
```

Dependency

```
use type_mod,      only : REKIND, DBKIND, INTKIND, TOKEN, STRING  
use axis_type_mod, only : AXISINFO, axis_type_copy  
use dc_trace,      only : SetDebug, BeginSub, EndSub, DbgMessage
```

Input

```
type(AXISINFO), intent(in)  :: axis
```

14.10.3 Set Variables

出力する変数を設定する。複数回呼ぶ事で複数の変数を設定できる。このサブルーチンで設定するのは変数キー varkey のみであり、具体的な情報は varinfo_mod の varinfo_init にて NAMELIST varinfo_nml で設定される。(本来は、デフォルトの値はプログラム無いにハードコードすべきかも知れない)。

```
subroutine io_gt4_out_SetVars(varkey)
```

Dependency

```
use type_mod,      only : REKIND, DBKIND, INTKIND, TOKEN, STRING  
use time_mod,      only : InitTime, DelTime, StepInterval, &  
  &                  tvar, ttype, tname, tunit  
use varinfo_mod,   only : varinfo_inquire, VAR_INFO  
use gt4_history,   only : GT_HISTORY , GT_HISTORY_AXIS , &
```

```

&          HistoryCreate, HistoryAddVariable , &
&          HistoryCopyVariable          , &
&          HistoryAddAttr, HistoryPut
use dc_string,    only : JoinChar
use dc_trace,    only : SetDebug, BeginSub, EndSub, DbgMessage

```

Input

```
character(*), intent(in)      :: varkey      ! 変数キー
```

14.10.4 Put 3-Dimensional Single Precision Data to netCDF file

変数キー varkey にデータ xyz_Var を出力する。varinfo_mod の varinfo_init の NAMELIST varinfo_nml で対応する varkey が与えられていない場合、データは出力されない。

各 varkey に対応する varinfo_mod の StepInterval および OutputStep と、time_mod の CurrentLoop から、出力するタイミングが正しいかどうかをチェックして出力する。出力するタイミングでないと判定された場合は何もせずに終了する。なお、varinfo_mod の StepInterval および OutputStep が無効な値 (ゼロ以下) の場合には time_mod の StepInterval と OutputStep が用いられる。具体的な判定方法は以下の通りである。

- CurrentLoop を StepInterval で割り、余りが 0 の場合には出力。
- CurrentLoop が StepInterval * OutputStep よりも大きくなってしまったら以降出力は行なわない。

```
subroutine io_gt4_out_Put3Real(varkey, xyz_Var)
```

Dependency

```

use type_mod,    only : REKIND, DBKIND, INTKIND, TOKEN, STRING
use time_mod,    only : StepInterval, OutputStep, CurrentLoop
use varinfo_mod, only : varinfo_inquire, VAR_INFO
use gt4_history, only : HistoryPut
use dc_trace,    only : SetDebug, BeginSub, EndSub, DbgMessage
use dc_message,  only : MessageNotify

```

Input

```

character(*),      intent(in):: varkey      ! 変数名
real(REKIND),      intent(in):: xyz_Var(:, :, :) ! 出力データ

```

14.10.5 Put 2-Dimensional Single Precision Data to netCDF file

機能は io_gt4_out_Put3Real と基本的に同じ。ただしこちらは単精度実数 2 次元のデータを出力する。

```
subroutine io_gt4_out_Put2Real(varkey, xy_Var)
```

Dependency

```
use type_mod,    only : REKIND, DBKIND, INTKIND, TOKEN, STRING
use time_mod,    only : StepInterval, OutputStep, CurrentLoop
use varinfo_mod, only : varinfo_inquire, VAR_INFO
use gt4_history, only : HistoryPut
use dc_trace,    only : SetDebug, BeginSub, EndSub, DbgMessage
use dc_message,  only : MessageNotify
```

Input

```
character(*),    intent(in):: varkey      ! 変数名
real(REKIND),    intent(in):: xy_Var(:,) ! 出力データ
```

14.10.6 Put 0-Dimensional Single Precision Data to netCDF file

機能は io_gt4_out_Put3Real と基本的に同じ。ただしこちらは単精度実数 0 次元のデータを出力する。

```
subroutine io_gt4_out_Put0Real(varkey, Var)
```

Dependency

```
use type_mod,    only : REKIND, DBKIND, INTKIND, TOKEN, STRING
use time_mod,    only : StepInterval, OutputStep, CurrentLoop
use varinfo_mod, only : varinfo_inquire, VAR_INFO
use gt4_history, only : HistoryPut
use dc_trace,    only : SetDebug, BeginSub, EndSub, DbgMessage
use dc_message,  only : MessageNotify
```

Input

```
character(*),    intent(in):: varkey      ! 変数名
real(REKIND),    intent(in):: Var        ! 出力データ
```

14.10.7 Put 3-Dimensional Double Precision Data to netCDF file

機能は io_gt4_out_Put3Real と基本的に同じ。ただしこちらは倍精度実数 3 次元のデータを出力する。

```
subroutine io_gt4_out_Put3Double(varkey, xyz_Var)
```

Dependency

```
use type_mod,    only : REKIND, DBKIND, INTKIND, TOKEN, STRING
use time_mod,    only : StepInterval, OutputStep, CurrentLoop
use varinfo_mod, only : varinfo_inquire, VAR_INFO
use gt4_history, only : HistoryPut
use dc_trace,    only : SetDebug, BeginSub, EndSub, DbgMessage
use dc_message,  only : MessageNotify
```

Input

```
character(*),    intent(in):: varkey          ! 変数名
real(DBKIND),    intent(in):: xyz_Var(:, :, :) ! 出力データ
```

14.10.8 Put 2-Dimensional Double Precision Data to netCDF file

機能は io_gt4_out_Put3Real と基本的に同じ。ただしこちらは倍精度実数 2 次元のデータを出力する。

```
subroutine io_gt4_out_Put2Double(varkey, xy_Var)
```

Dependency

```
use type_mod,    only : REKIND, DBKIND, INTKIND, TOKEN, STRING
use time_mod,    only : StepInterval, OutputStep, CurrentLoop
use varinfo_mod, only : varinfo_inquire, VAR_INFO
use gt4_history, only : HistoryPut
use dc_trace,    only : SetDebug, BeginSub, EndSub, DbgMessage
use dc_message,  only : MessageNotify
```

Input

```
character(*),    intent(in):: varkey          ! 変数名
real(DBKIND),    intent(in):: xy_Var(:, :)   ! 出力データ
```

14.10.9 Put 2-Dimensional Double Precision Data to netCDF file

機能は io_gt4_out_Put3Real と基本的に同じ。ただしこちらは倍精度実数 2 次元のデータを出力する。

```
subroutine io_gt4_out_Put0Double(varkey, Var)
```

Dependency

```
use type_mod,    only : REKIND, DBKIND, INTKIND, TOKEN, STRING
use time_mod,    only : StepInterval, OutputStep, CurrentLoop
use varinfo_mod, only : varinfo_inquire, VAR_INFO
use gt4_history, only : HistoryPut
use dc_trace,    only : SetDebug, BeginSub, EndSub, DbgMessage
use dc_message,  only : MessageNotify
```

Input

```
character(*),    intent(in):: varkey      ! 変数名
real(DBKIND),    intent(in):: Var        ! 出力データ
```

14.10.10 Terminate module

io_gt4_out_init で設定された値を破棄し、デフォルトの値に戻す。また、HistoryClose によって、HistoryCreate 等に対応する終了処理を行なう。

```
subroutine io_gt4_out_end
```

Dependency

```
use type_mod,    only : REKIND, DBKIND, INTKIND, TOKEN, STRING
use gt4_history, only : HistoryClose
use dc_trace,    only : SetDebug, BeginSub, EndSub, DbgMessage
```

15 Module dynamics_mod

- Developers: Morikawa Yasuhiro
- Version: \$Id: dynamics.f90,v 1.20 2005/01/22 09:30:59 morikawa Exp \$
- Tag Name: \$Name: \$
- Change History:

15.1 Overview

Calculate Dynamical Core.

力学コア部分を演算するモジュール。演算している方程式系、および離散化の手法は以下の通り。

- 支配方程式系 : Governing Equations
 - 球座標プリミティブ方程式系 : Primitive Equations in Spherical Coordinate
- 水平離散化 : Horizontal Discretization
 - スペクトル法 : Spectral Method
 - * 三角形切断 : Triangle Truncation
 - * 変換法 : Transform Method
- 鉛直離散化 : Vertical Discretization
 - 座標系 : Sigma Coordinate (Arakawa and Suarez(1983))
 - * Lorenz 格子 : Lorenz Grid
- 時間積分 : Time Integral
 - leap frog : リープフロッグ

15.2 Reference

- Arakawa, A., Suarez, M. J., 1983: Vertical differencing of the primitive equations in sigma coordinates. Mon. Wea. Rev., 111, 34–35.

15.3 Error Handling

15.4 Known Bugs

- 粘性計算の `wa_NumVis_wa` と `wa_NumVisScaler_wa` をとりあえず SP-MODEL から移植したため、名称などがそのままである。本来はモジュール内部のものとして名称その他の検討が必要であろう。

15.5 Note

15.6 Future Plans

現在、力学コアの全てがこのモジュール内にある。本来ならば、解く方程式や項の種類によってモジュール化がなされるべきである。

15.7 Dependency

```
use type_mod,    only: STRING, REKIND, DBKIND, INTKIND
```

15.8 Public Interface

```
private
public :: dynamics_init, dynamics_leapfrog ! subroutines
public :: dynamics_diagnostic, dynamics_end ! subroutines
public :: dynamics_diffusion             ! subroutines
```

15.9 Procedure Interface

15.9.1 Initialize module and Calculate Non-Predictional Value.

以降のサブルーチンで用いる変数の allocate、および時間発展しない量の演算を行なう。また、変数データ出力のための初期設定も行なう。

```
subroutine dynamics_init(x_Lon, y_Lat, z_Sigma, r_Sigma)
```

Dependency

```
use type_mod,    only: STRING, REKIND, DBKIND, INTKIND
use grid_3d_mod,    only: im, jm, km
use grid_wavenumber_mod, only: nm
use constants_mod, only: constants_init, R0, Omega, Cp, RAir, &
    &                TempAve, VisOrder, EFoldTime
use time_mod,    only: DelTime
use spml_mod,    only: spml_init, xy_Lat, rn
use io_gt4_out_mod, only: io_gt4_out_init, io_gt4_out_SetVars
use dc_trace,    only: DbgMessage, BeginSub, EndSub, DataDump
use dc_string,   only: toChar
```

Input

```
real(DBKIND), intent(in) :: &
    & x_Lon(:)           , & ! intent(in): 経度座標
    & y_Lat(:)           , & ! intent(in): 緯度座標
    & z_Sigma(:)         , & ! intent(in): レベル(整数)座標
    & r_Sigma(:)         ! intent(in): レベル(半整数)座標
```

標

15.9.2 Calculate Predictional Values.

時間発展する量の演算を行なう。演算するデータの出力も行なう。

```
subroutine dynamics_leapfrog                                     &
    & ( x_Lon           , y_Lat           , z_Sigma   , r_Sigma   , &
    &
    & xyz_VelLon_b, xyz_VelLat_b, xyz_Vor_b, xyz_Div_b , &
    & xyz_Temp_b  , xyz_QVap_b  , xy_Ps_b  ,           &
    &
    & xyz_VelLon_n, xyz_VelLat_n, xyz_Vor_n, xyz_Div_n , &
    & xyz_Temp_n  , xyz_QVap_n  , xy_Ps_n  ,           &
    &
    & xyz_VelLon_a, xyz_VelLat_a, xyz_Vor_a, xyz_Div_a , &
    & xyz_Temp_a  , xyz_QVap_a  , xy_Ps_a  )
```

Dependency

```
use type_mod,    only: STRING, REKIND, DBKIND, INTKIND
use grid_3d_mod,    only: im, jm, km
use grid_wavenumber_mod, only: nm
use constants_mod, only: R0, Cp, EpsVT
use time_mod,     only: DelTime, CurrentTime
use spml_mod,     only: w_xy, xy_w , xy_GradLon_w, xy_GradLat_w, &
    &                w_Div_xy_xy, w_LaplaInv_w,           &
    &                wa_xya, xya_wa, wa_Div_xya_xya, wa_Lapla_wa
use io_gt4_out_mod, only: io_gt4_out_Put
use dc_trace,     only: DbgMessage, BeginSub, EndSub, DataDump
use dc_string,   only: toChar
```

Input

```

real(DBKIND), intent(in) :: &
    & x_Lon(:)           , & ! intent(in): 経度座標
    & y_Lat(:)           , & ! intent(in): 緯度座標
    & z_Sigma(:)         , & ! intent(in): レベル(整数)座標
    & r_Sigma(:)         , & ! intent(in): レベル(半整数)座標
    &
    & xyz_VelLon_b(:,:,:) , & ! intent(in): 速度経度成分 (t-
t)
    & xyz_VelLat_b(:,:,:) , & ! intent(in): 速度緯度成分 (t-
t)
    & xyz_Vor_b(:,:,:)   , & ! intent(in): 渦度           (t-
t)
    & xyz_Div_b(:,:,:)   , & ! intent(in): 発散           (t-
t)
    & xyz_Temp_b(:,:,:)  , & ! intent(in): 温度           (t-
t)
    & xyz_QVap_b(:,:,:)  , & ! intent(in): 比湿           (t-
t)
    & xy_Ps_b(:,:)       , & ! intent(in): 地表面気圧      (t-
t)
    &
    & xyz_VelLon_n(:,:,:) , & ! intent(in): 速度経度成分 (t)
    & xyz_VelLat_n(:,:,:) , & ! intent(in): 速度緯度成分 (t)
    & xyz_Vor_n(:,:,:)   , & ! intent(in): 渦度           (t)
    & xyz_Div_n(:,:,:)   , & ! intent(in): 発散           (t)
    & xyz_Temp_n(:,:,:)  , & ! intent(in): 温度           (t)
    & xyz_QVap_n(:,:,:)  , & ! intent(in): 比湿           (t)
    & xy_Ps_n(:,:)       ! intent(in): 地表面気圧      (t)

```

Output

```

real(DBKIND), intent(out) :: &
    & xyz_VelLon_a(:,:,:) , & ! intent(out): 速度経度成分 (t+
t)
    & xyz_VelLat_a(:,:,:) , & ! intent(out): 速度緯度成分 (t+
t)
    & xyz_Vor_a(:,:,:)   , & ! intent(out): 渦度           (t+ t)
    & xyz_Div_a(:,:,:)   , & ! intent(out): 発散           (t+ t)
    & xyz_Temp_a(:,:,:)  , & ! intent(out): 温度           (t+ t)
    & xyz_QVap_a(:,:,:)  , & ! intent(out): 比湿           (t+ t)

```

```
& xyz_Ps_a(:, :)          ! intent(out): 地表面気圧 (t+ t)
```

15.9.3 Calculate Diffusion Term

t- t の値から水平拡散項を求め、それを t+ t の値に加える。

```
subroutine dynamics_diffusion(                                     &
  & xyz_Vor_b , xyz_Div_b , xyz_Temp_b , xyz_QVap_b , &
  & xyz_Vor_a , xyz_Div_a , xyz_Temp_a , xyz_QVap_a          )
```

Dependency

```
use type_mod,      only: STRING, REKIND, DBKIND, INTKIND
use time_mod,     only: DelTime, CurrentTime
use grid_3d_mod,  only: km
use grid_wavenumber_mod, only: nm
use spml_mod,     only: wa_xya, xya_wa, l_nm
use io_gt4_out_mod, only: io_gt4_out_Put
use dc_trace,     only: DbgMessage, BeginSub, EndSub, DataDump
```

Input

```
real(DBKIND), intent(in) :: &
  & xyz_Vor_b(:, :, :) , & ! 渦度 (t- t)
  & xyz_Div_b(:, :, :) , & ! 発散 (t- t)
  & xyz_Temp_b(:, :, :) , & ! 温度 (t- t)
  & xyz_QVap_b(:, :, :) ! 比湿 (t- t)
```

In/Out

```
real(DBKIND), intent(inout) :: &
  & xyz_Vor_a(:, :, :) , & ! 渦度 (t+ t)
  & xyz_Div_a(:, :, :) , & ! 発散 (t+ t)
  & xyz_Temp_a(:, :, :) , & ! 温度 (t+ t)
  & xyz_QVap_a(:, :, :) ! 比湿 (t+ t)
```

15.9.4 Calculate Diagnostic Values.

診断的に得られる量の演算を行なう。現在は渦度発散から速度成分 (経度方向、緯度方向) の演算を行なうのみである。

```

subroutine dynamics_diagnostic                                     &
  & ( x_Lon           , y_Lat           , z_Sigma   , r_Sigma   , &
  &
  & xyz_VelLon_a, xyz_VelLat_a, xyz_Vor_a, xyz_Div_a , &
  & xyz_Temp_a   , xyz_QVap_a   , xy_Ps_a       )

```

Dependency

```

use type_mod,      only: STRING, REKIND, DBKIND, INTKIND
use grid_3d_mod,   only: im, jm, km
use grid_wavenumber_mod, only: nm
use constants_mod, only: R0, Grav
use spml_mod,      only: wa_xya, xya_GradLon_wa, xya_GradLat_wa, &
  &
  & wa_LaplaInv_wa, IntLonLat_xy
use io_gt4_out_mod, only: io_gt4_out_Put
use dc_trace,      only: DbgMessage, BeginSub, EndSub, DataDump
use dc_string,     only: toChar

```

Input

```

real(DBKIND), intent(in) :: &
  & x_Lon(:)           , & ! intent(in): 経度座標
  & y_Lat(:)           , & ! intent(in): 緯度座標
  & z_Sigma(:)         , & ! intent(in): レベル(整数)座標
  & r_Sigma(:)         , & ! intent(in): レベル(半整数)座標
  &
  & xyz_Vor_a(:,:,:)  , & ! intent(in): 渦度           (t+ t)
  & xyz_Div_a(:,:,:)  , & ! intent(in): 発散           (t+ t)
  & xyz_Temp_a(:,:,:) , & ! intent(in): 温度           (t+ t)
  & xyz_QVap_a(:,:,:) , & ! intent(in): 比湿           (t+ t)
  & xy_Ps_a(:,:)      ! intent(in): 地表面気圧 (t+ t)

```

In/Output

```

real(DBKIND), intent(out) :: &
  & xyz_VelLon_a(:,:,:) , & ! intent(out): 速度経度成分 (t+
t)
  & xyz_VelLat_a(:,:,:) ! intent(out): 速度緯度成分 (t+
t)

```

15.9.5 Terminate module

dynamics_init で allocate した変数を deallocate し、演算した値も全て破棄する。

```
subroutine dynamics_end
```

Dependency

```
use type_mod,    only: STRING, REKIND, DBKIND, INTKIND
use dc_trace,    only: BeginSub, EndSub, DbgMessage
```

16 Module hs94forcing_mod

- Developers: Morikawa Yasuhiro
- Version: \$Id: hs94forcing.f90,v 1.9 2005/01/22 09:23:41 morikawa Exp \$
- Tag Name: \$Name: \$
- Change History:

16.1 Overview

This module compute heating and dissipation for Held and Suarez(1994) benchmark integration of a dry GCM.

Held and Suarez(1994) の乾燥大気 GCM ベンチマーク用の加熱と散逸を計算する。

16.2 Reference

- Held, I. M., Suarez, M. J., 1994: A proposal for the intercomparison of the dynamical cores of atmospheric general circulation models. Bull. Am. Meteor. Soc., 75, 1825–1830.

16.3 Error Handling

16.4 Known Bugs

- 地球半径や回転角速度、重力加速度などの物理定数が constant_mod に依存するようになっている。正しくテストしたいのならば、それらが

Held and Suarez(1994) の実験設定と整合的かどうかチェックするよう
にすべきである。

16.5 Note

16.6 Future Plans

16.7 Dependency

```
use type_mod,      only : STRING, DBKIND
```

16.8 Public Interface

```
private
public:: hs94forcing_init, hs94forcing, hs94forcing_end ! subroutines
```

Input

```
type(AxisInfo), intent(in) :: &
    & x_Lon           , & ! 経度座標
    & y_Lat           , & ! 緯度座標
    & z_Sigma         !   レベル (整数) 座標
```

16.8.1 加熱と散逸の計算

加熱と散逸の結果は xyz_VelLon_phy, xyz_VelLat_phy, xyz_Temp_phy と
して返る。

```
subroutine hs94forcing( &
    & xyz_VelLon_b   , xyz_VelLat_b   , xyz_Temp_b   , xy_Ps_b   , &
    & xyz_VelLon_phy, xyz_VelLat_phy, xyz_Temp_phy   )
```

Dependency

```
use type_mod,      only: INTKIND, STRING, TOKEN, REKIND, DBKIND
use constants_mod, only: RAir, Cp, SecPerDay
use grid_3d_mod   , only: im, jm, km
use io_gt4_out_mod, only: io_gt4_out_Put
use dc_trace      , only: DbgMessage, BeginSub, EndSub, DataDump
```


Input

```
real(DBKIND), intent(in) :: &
    & xyz_VelLon_b(:,:,:) , & ! 速度経度成分 (t- t)
    & xyz_VelLat_b(:,:,:) , & ! 速度緯度成分 (t- t)
    & xyz_Temp_b(:,:,:)   , & ! 温度          (t- t)
    & xy_Ps_b(:,:)       ! 地表面気圧   (t- t)
```

Output

```
real(DBKIND), intent(out) :: &
    & xyz_VelLon_phy(:,:,:) , & ! 速度経度成分の加熱散逸効果
    & xyz_VelLat_phy(:,:,:) , & ! 速度緯度成分の加熱散逸効果
    & xyz_Temp_phy(:,:,:)   ! 温度の加熱散逸効果
```

16.8.2 Terminate module

hs94forcing_init で allocate した変数を deallocate する。

```
subroutine hs94forcing_end
```

Dependency

```
use type_mod, only: STRING, DBKIND, INTKIND
use dc_trace, only: BeginSub, EndSub, DbgMessage
```