地球流体電脳ライブラリ

数学関数

(Fortran 90 版)

地球流体電脳倶楽部

2015 年 7 月 11 日

目次

第 1 章

内部変数管理

1.1 ルーチンリスト

DclGetParm	内部変数を参照する.
(???GET)	
DclSetParm	内部変数を変更する.
(???SET)	
DclSetParmEx	内部変数を変更する (実行時オプションによる変更を許す).
(???STX)	

* 括弧の中は, 対応する f77 インターフェイス名.

1.2 各ルーチンの説明

DclGetParm

1. 機能

   内部変数を参照する.

2. 書式

   ```
   call DclGetParm(name, value)
   ```

3. 引数

name	<C>	内部変数の名前.
value	<I,RC,L>	内部変数の値.

4. 備考

   - 内部変数の名前はクォーテーション (' ') でくくって用いる. 内部変数の名前, 型についてはパラメータリストを参照のこと.

5. 関連項目

   - 関連ルーチン (内部変数管理)
   - 内部変数
   - パラメータリスト

DclSetParm

1. 機能

   内部変数を変更する.

2. 書式

   ```
   call DclSetParm(name, value)
   ```

3. 引数

name	<C>	内部変数の名前.
value	<I,RC,L>	内部変数の値.

4. 備考

   - 内部変数の名前はクォーテーション (' ') でくくって用いる. 内部変数の名前, 型についてはパラメータリストを参照のこと.

5. 関連項目

   - 関連ルーチン (内部変数管理)
   - 内部変数
   - パラメータリスト

DclSetParmEx

1. 機能

   内部変数を変更する (実行時オプションによる変更を許す).

2. 書式

   ```
   call DclSetParmEx(name, value)
   ```

3. 引数

name	<C>	内部変数の名前.
value	<I,RC,L>	内部変数の値.

4. 備考

   内部変数の名前はクォーテーション (' ') でくくって用いる. 内部変数の名前, 型についてはパラメータリストを参照のこと.

   内部変数の設定手段を, その効力の強さの順に並べると,

   (a) DclSetParm

   (b) 実行時オプション

   (c) DclSetParmEx

   (d) システムデフォルト

   となる.

5. 関連項目

   - 関連ルーチン (内部変数管理)
   - 内部変数
   - パラメータリスト

第 2 章

システム管理

2.1 ルーチンリスト

DclMessageDump	メッセージを出力する.
(MSGDMP)	
DclCompChar	大文字・小文字の区別なく
(LCHREQ)	文字列を比較する.
DclGetUnitNum	利用可能なもっとも小さい
(IUFOPN)	入出力装置番号を返す.

* 括弧の中は, 対応する f77 インターフェイス名.

2.2 各ルーチンの説明

2.2.1 DclMessageDump

1. 機能

   メッセージを出力する.

2. 書式

   ```
   call DclMessageDump(level, name, message)
   ```

3. 引数

level	<C1>	メッセージレベル.
		'E', 'W', 'M'.
name	<C32>	エラーメッセージを出すルーチン名.
message	<C*>	メッセージ.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (syslib)

2.2.2 DclGetUnitNum

1. 機能

   利用可能なもっとも小さい入出力装置番号を返す.

2. 書式

   ```
   result=DclGetUnitNum()
   ```

3. 引数

   戻り値 <I> 入出力装置番号.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (syslib)

2.2.3 DclCompChar

1. 機能

   大文字・小文字の区別なく文字列を比較する.

2. 書式

   ```
   result=DclCompChar(ch1, ch2)
   ```

3. 引数

   戻り値 <L> 文字列が等しい時 .true. を返す.

   ch1, ch2 <C*> 比較する文字列.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (syslib)

第 3 章

システム依存

3.1 ルーチンリスト

DclExecCommand	os コマンドを実行する.
(OSEXEC)	
DclGetEnv	環境変数の値を取得する.
(OSGENV)	
DclGetArgumentNum	コマンドライン引数の数 N を得る.
(OSQARN)	
DclGetArgument	N 番目のコマンドライン引数を得る.
(OSGARG)	
DclAbort	エラー処理をおこなって強制終了する.
(OSABRT)	

* 括弧の中は, 対応する f77 インターフェイス名.

3.2 各ルーチンの説明

3.2.1 DclExecCommand

1. 機能

   os コマンドを実行する.

2. 書式

   ```
   call DclExecCommand(command)
   ```

3. 引数

command	<C*>	コマンド文字列.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (oslib)

3.2.2 DclGetEnv

1. 機能

   環境変数の値を取得する.

2. 書式

   ```
   call DclGetEnv(name, value)
   ```

3. 引数

name	<C*>	環境変数名.
value	<C*>	環境変数の値.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (oslib)

3.2.3 DclGetArgumentNum

1. 機能

   コマンドライン引数の数 N を得る.

2. 書式

   ```
   result=DclGetArgumentNum()
   ```

3. 引数

戻り値	<I>	コマンドライン引数の数.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (oslib)

3.2.4 DclGetArgument

1. 機能

   N 番目のコマンドライン引数を得る.

2. 書式

   ```
   call DclGetArgument(n, arg)
   ```

3. 引数

n	<I>	コマンドライン引数の順番.
arg	<C*>	引数の値.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (oslib)

3.2.5 DclAbort

1. 機能

   エラー処理をおこなって強制終了する.

2. 書式

   ```
   result=DclAbort()
   ```

3. 引数なし.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (oslib)

第 4 章

大文字小文字変換

4.1 ルーチンリスト

`DclToUpper` 文字列を大文字化する.
`(CUPPER)`
`DclToLower` 文字列を小文字化する.
`(CLOWER)`

* 括弧の中は, 対応する f77 インターフェイス名.

4.2 各ルーチンの説明

4.2.1 DclToUpper

1. 機能
   文字列を大文字化する.
2. 書式
   ```
   call DclToUpper(ch)
   ```
3. 引数
   `ch` `<C*>` 変換する文字列.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (chglib)

4.2.2 DclToLower

1. 機能
   文字列を小文字化する.
2. 書式
   ```
   call DclToLower(ch)
   ```

3. 引数

   `ch` `<C*>` 変換する文字列.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (chglib)

第 5 章

文字種の判別

5.1 ルーチンリスト

DclCheckBlank	空白かどうかを判別する.
(LCHRB)	
DclCheckCurrency	通貨記号かどうかを判別する.
(LCHRC)	
DclCheckSpecial	特殊文字かどうかを判別する.
(LCHRS)	
DclCheckAlphabet	英字かどうかを判別する.
(LCHRL)	
DclCheckNumber	数字かどうかを判別する.
(LCHRD)	
DclCheckAlphaNum	英数字かどうかを判別する.
(LCHRA)	
DclCheckFortran	FORTRAN 文字かどうかを判別する.
(LCHRF)	
DclCheckCharPattern	指定した文字種かどうかを判別する.
(LCHR)	

* 括弧の中は, 対応する f77 インターフェイス名.

5.2 各ルーチンの説明

5.2.1 DclCheckBlank

1. 機能

   空白かどうかを判別する.

2. 書式

   ```
   result=DclCheckBlank(c)
   ```

3. 引数

戻り値 <L> 判定結果.
c <C1> 判定する文字.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (chklib)

5.2.2 DclCheckCurrency

1. 機能

通貨記号かどうかを判別する.

2. 書式

```
result=DclCheckCurrency(c)
```

3. 引数

戻り値 <L> 判定結果.
c <C1> 判定する文字.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (chklib)

5.2.3 DclCheckSpecial

1. 機能

特殊文字かどうかを判別する.

2. 書式

```
result=DclCheckSpecial(c)
```

3. 引数

戻り値 <L> 判定結果.
c <C1> 判定する文字.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (chklib)

5.2.4 DclCheckAlphabet

1. 機能

英字かどうかを判別する.

2. 書式

```
result=DclCheckAlphabet(c)
```

3. 引数

戻り値	<L>	判定結果.
c	<C1>	判定する文字.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (chklib)

5.2.5 DclCheckNumber

1. 機能

数字かどうかを判別する.

2. 書式

```
result=DclCheckNumber(c)
```

3. 引数

戻り値	<L>	判定結果.
c	<C1>	判定する文字.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (chklib)

5.2.6 DclCheckAlphaNum

1. 機能

英数字かどうかを判別する.

2. 書式

```
result=DclCheckAlphaNum(c)
```

3. 引数

戻り値	<L>	判定結果.
c	<C1>	判定する文字.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (chklib)

5.2.7 DclCheckFortran

1. 機能

FORTRAN 文字かどうかを判別する.

2. 書式

```
result=DclCheckFortran(c)
```

3. 引数

戻り値	<L>	判定結果.
c	<C1>	判定する文字.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (chklib)

5.2.8 DclCheckCharPattern

1. 機能

指定した文字種かどうかを判別する.

2. 書式

```
result=DclCheckCharPattern(char, cref)
```

3. 引数

戻り値	<L>	判定結果.
char	<C*>	調べる文字列.
cref	<C*>	テンプレート.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (chklib)

第 6 章

日付

6.1 ルーチンリスト

DclGetDate	今日の日付を求める.
(DATEQ1,2,3)	
DclAddDate	日付に日数を加える.
(DATEF1,2,3)	
DclDiffDate	日付の差を求める.
(DATEG1,2,3)	
DclFormatDate	日付を指定された
(DATEC1,2,3)	フォーマットに変換する.
DclDayOfWeek	曜日番号を求める.
(IWEEK1,2,3)	
DclLengthOfMonth	月の長さ (日数) を求める.
(NDMON)	
DclLengthOfYear	1 年の長さ (日数) を求める.
(NDYEAR)	

* 括弧の中は, 対応する f77 インターフェイス名.

6.2 各ルーチンの説明

6.2.1 DclGetDate

1. 機能

   今日の日付を求める.

2. 書式

   ```
   result=DclGetDate()
   ```

3. 引数

   戻り値　<dcl_date>　今日の日付.

4. 備考

なし.

5. 関連項目
   - 関連ルーチン (datelib)

6.2.2 DclAddDate

1. 機能

   日付に日数を加える.

2. 書式

```
result=DclAddDate(date, n)
```

3. 引数

戻り値	<dcl_date>	今日の日付.
date	<dcl_date>	もとの日付.
n	<I>	加える日数.

4. 備考

   なし.

5. 関連項目
   - 関連ルーチン (datelib)

6.2.3 DclDiffDate

1. 機能

   日付の差を求める.

2. 書式

```
result=DclDiffDate(date1, date2)
```

3. 引数

戻り値	<I>	求める日数.
date1, date2	<dcl_date>	日付.

4. 備考

   なし.

5. 関連項目
   - 関連ルーチン (datelib)

6.2.4 DclFormatDate

1. 機能

   日付を指定されたフォーマットに変換する.

2. 書式

```
call DclFormatDate(cform, date)
```

3. 引数

cform	<C*>	フォーマット. 指定されたところに日付データが埋めこまれて出力される.
date	<dcl_date>	日付.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (datelib)

6.2.5 DclDayOfWeek

1. 機能

   曜日番号を求める.

2. 書式

```
result=DclDayOfWeek(date)
```

3. 引数

戻り値	<I>	曜日番号.
date	<dcl_date>	日付.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (datelib)

6.2.6 DclLengthOfMonth

1. 機能

   月の長さ (日数) を求める.

2. 書式

```
result=DclLengthOfMonth(year, month)
```

3. 引数

戻り値	<I>	月の長さ (日数).
year	<I>	年.
month	<I>	月.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (datelib)

6.2.7 DclLengthOfYear

1. 機能

   1 年の長さ (日数) を求める.

2. 書式

```
result=DclLengthOfYear(year)
```

3. 引数

戻り値 <I> 1 年の長さ (日数).

year <I> 年.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (datelib)

第 7 章

時間

7.1 ルーチンリスト

DclGetTime	現在の時刻を求める.
(TIMEQ1,2,3)	
DclFormatTime	時刻を指定された
(TIMEC1,2,3)	フォーマットに変換する.

* 括弧の中は, 対応する f77 インターフェイス名.

7.2 各ルーチンの説明

7.2.1 DclGetTime

1. 機能
   現在の時刻を求める.
2. 書式
   ```
   result=DclGetTime()
   ```
3. 引数
   戻り値　<dcl_time>　現在の時刻.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (timelib)

7.2.2 DclFormatTime

1. 機能
   時刻を指定されたフォーマットに変換する.
2. 書式
   ```
   call DclFormatTime(cform, time)
   ```

3. 引数

`cform`	`<C*>`	フォーマット. 指定されたところに時刻情報が埋めこまれて出力される.
`time`	`<dcl_time>`	時刻.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (timelib)

第 8 章

実数値の誤差を含む比較

8.1 ルーチンリスト

DclEQ	指定された誤差の範囲で,
(LREQ,LREQA)	x と y が等しいかどうか調べる.
DclNE	指定された誤差の範囲で,
(LRNE,LRNEA)	x と y が等しくないかどうか調べる.
DclLT	指定された誤差の範囲で,
(LRLT,LRLTA)	x が y より小さいかどうか調べる.
DclLE	指定された誤差の範囲で,
(LRLE,LRLEA)	x が y 以下かどうか調べる.
DclGT	指定された誤差の範囲で,
(LRGT,LRGTA)	x が y より大きいかどうか調べる.
DclGE	指定された誤差の範囲で,
(LRGE,LRGEA)	x が y 以上かどうか調べる.

* 括弧の中は, 対応する f77 インターフェイス名.

8.2 各ルーチンの説明

8.2.1 DclEQ

1. 機能

   指定された誤差の範囲で, x と y が等しいかどうか調べる.

2. 書式

   ```
   result=DclEQ(x, y, [epsilon])
   ```

3. 引数

戻り値	<L>	判定結果. x と y が等しい時に .true.
x, y	<R>	比較する実数.
epsilon	<R>	許容誤差. 省略時は xxx.

4. 備考

   なし.

5. 関連項目
   - 関連ルーチン (lrllib)

8.2.2 DclNE

1. 機能
   指定された誤差の範囲で, x と y が等しくないかどうか調べる.
2. 書式
   ```
   result=DclNE(x, y, [epsilon])
   ```
3. 引数
   戻り値 <L> 判定結果. x と y が等しくない時に .true.
   x, y <R> 比較する実数.
   epsilon <R> 許容誤差. 省略時は xxx.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (lrllib)

8.2.3 DclLT

1. 機能
   指定された誤差の範囲で, x が y より小さいかどうか調べる.
2. 書式
   ```
   result=DclLT(x, y, [epsilon])
   ```
3. 引数
   戻り値 <L> 判定結果. x が y より小さい時に .true.
   x, y <R> 比較する実数.
   epsilon <R> 許容誤差. 省略時は xxx.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (lrllib)

8.2.4 DclLE

1. 機能
   指定された誤差の範囲で, x が y 以下かどうか調べる.
2. 書式
   ```
   result=DclLE(x, y, [epsilon])
   ```
3. 引数

戻り値	<L>	判定結果. x が y 以下の時に .true.
x, y	<R>	比較する実数.
epsilon	<R>	許容誤差. 省略時は xxx.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (lrllib)

8.2.5 DclGT

1. 機能

   指定された誤差の範囲で, x が y より大きいかどうか調べる.

2. 書式

```
result=DclGT(x, y, [epsilon])
```

3. 引数

戻り値	<L>	判定結果. x が y より大きい時に .true.
x, y	<R>	比較する実数.
epsilon	<R>	許容誤差. 省略時は xxx.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (lrllib)

8.2.6 DclGE

1. 機能

   指定された誤差の範囲で, x が y 以上かどうか調べる.

2. 書式

```
result=DclGE(x, y, [epsilon])
```

3. 引数

戻り値	<L>	判定結果. x が y 以上の時に .true.
x, y	<R>	比較する実数.
epsilon	<R>	許容誤差. 省略時は xxx.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (lrllib)

第 9 章

区分順序

9.1 ルーチンリスト

DclIntervalLT	bounds(i-1) ≦value <bounds(i)
(IBLKLT)	を満たす i を求める.
DclIntervalLE	bounds(i-1) <value ≦bounds(i)
(IBLKLE)	を満たす i を求める.
DclIntervalGT	bounds(i) <value ≦bounds(i+1)
(IBLKGT)	を満たす i を求める.
DclIntervalGE	bounds(i) ≦value <bounds(i+1)
(IBLKGE)	を満たす i を求める.

* 括弧の中は, 対応する f77 インターフェイス名.

9.2 各ルーチンの説明

9.2.1 DclIntervalGE

1. 機能

   bounds(i) ≦value < bounds(i+1) を満たす i を求める.

2. 書式

   result=DclIntervalGE(bounds, value)

3. 引数

戻り値	<I>	条件を満たす配列番号 i.
bounds	<R(:)>	指定する区分境界値.
value	<R>	指定する値.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (blklib)

9.2.2 DclIntervalGT

1. 機能

   bounds(i) <value ≤ bounds(i+1) を満たす i を求める.

2. 書式

   result=DclIntervalGT(bounds, value)

3. 引数

戻り値	<I>	条件を満たす配列番号 i.
bounds	<R(:)>	指定する区分境界値.
value	<R>	指定する値.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (blklib)

9.2.3 DclIntervalLE

1. 機能

   bounds(i-1) <value ≤ bounds(i) を満たす i を求める.

2. 書式

   result=DclIntervalLE(bounds, value)

3. 引数

戻り値	<I>	条件を満たす配列番号 i.
bounds	<R(:)>	指定する区分境界値.
value	<R>	指定する値.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (blklib)

9.2.4 DclIntervalLT

1. 機能

   bounds(i-1) ≤value < bounds(i) を満たす i を求める.

2. 書式

   result=DclIntervalLT(bounds, value)

3. 引数

戻り値	<I>	条件を満たす配列番号 i.
bounds	<R(:)>	指定する区分境界値.
value	<R>	指定する値.

4. 備考

なし.

5. 関連項目

   - 関連ルーチン (blklib)

第 10 章

きりの良い数

10.1　ルーチンリスト

■簡易型関数

DclGoodNumLT	指定値より小さい「きりのよい数」
(RGNLT)	のうちで最大のものを求める.
DclGoodNumGT	指定値より大きい「きりのよい数」
(RGNGT)	のうちで最小のものを求める.
DclGoodNumLE	指定値以下の「きりのよい数」
(RGNLE)	のうちで最大のものを求める.
DclGoodNumGE	指定値以上の「きりのよい数」
(RGNGE)	のうちで最小のものを求める.

■指数部，仮数部を返すサブルーチン

DclGoodNumExLT	指定値より小さい「きりのよい数」
(GNLT)	のうちで最大のものを求める.
DclGoodNumExGT	指定値より大きい「きりのよい数」
(GNGT)	のうちで最小のものを求める.
DclGoodNumExLE	指定値以下の「きりのよい数」
(GNLE)	のうちで最大のものを求める.
DclGoodNumExGE	指定値以上の「きりのよい数」
(GNGE)	のうちで最小のものを求める.

■設定・参照

DclSetGoodNumList	「きりのよい数」
(GNSBLK)	のリストを設定する.
DclGetGoodNumList	現在設定されている「きりのよい数」
(GNQBLK)	のリストを参照する.
DclSaveGoodNumList	現在設定されている「きりのよい数」
(GNSAVE)	のリストを保存する.
DclRestoreGoodNumList	保存されたリストで
(GNRSET)	「きりのよい数」を再設定する.

* 括弧の中は, 対応する f77 インターフェイス名.

10.2 各ルーチンの説明

10.2.1 DclGoodNumExLT

1. 機能

   指定値より小さい「きりのよい数」のうちで最大のものを求める.

2. 書式

```
call DclGoodNumExLT(value, mantissa, exponent )
```

3. 引数

value	<R>	指定値.
mantissa	<R>	きりの良い数の仮数部.
exponent	<I>	きりの良い数の指数部.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (gnmlib)

10.2.2 DclGoodNumExGT

1. 機能

   指定値より大きい「きりのよい数」のうちで最小のものを求める.

2. 書式

```
call DclGoodNumExGT(value, mantissa, exponent )
```

3. 引数

value	<R>	指定値.
mantissa	<R>	きりの良い数の仮数部.
exponent	<I>	きりの良い数の指数部.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (gnmlib)

10.2.3 DclGoodNumExLE

1. 機能

   指定値以下の「きりのよい数」のうちで最大のものを求める.

2. 書式

   ```
   call DclGoodNumExLE(value, mantissa, exponent )
   ```

3. 引数

value	<R>	指定値.
mantissa	<R>	きりの良い数の仮数部.
exponent	<I>	きりの良い数の指数部.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (gnmlib)

10.2.4 DclGoodNumExGE

1. 機能

   指定値以上の「きりのよい数」のうちで最小のものを求める.

2. 書式

   ```
   call DclGoodNumExGE(value, mantissa, exponent )
   ```

3. 引数

value	<R>	指定値.
mantissa	<R>	きりの良い数の仮数部.
exponent	<I>	きりの良い数の指数部.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (gnmlib)

10.2.5 DclGoodNumLT

1. 機能

   指定値より小さい「きりのよい数」のうちで最大のものを求める.

2. 書式

   ```
   result=DclGoodNumLT(value)
   ```

3. 引数

戻り値	<R>	きりの良い数.
value	<R>	指定値.

4. 備考

   なし.

5. 関連項目
   - 関連ルーチン (gnmlib)

10.2.6 DclGoodNumGT

1. 機能
   指定値より大きい「きりのよい数」のうちで最小のものを求める.
2. 書式
   ```
   result=DclGoodNumGT(value)
   ```
3. 引数
   戻り値 <R> きりの良い数.
   value <R> 指定値.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (gnmlib)

10.2.7 DclGoodNumLE

1. 機能
   指定値以下の「きりのよい数」のうちで最大のものを求める.
2. 書式
   ```
   result=DclGoodNumLE(value)
   ```
3. 引数
   戻り値 <R> きりの良い数.
   value <R> 指定値.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (gnmlib)

10.2.8 DclGoodNumGE

1. 機能
   指定値以上の「きりのよい数」のうちで最小のものを求める.
2. 書式
   ```
   result=DclGoodNumGE(value)
   ```
3. 引数
   戻り値 <R> きりの良い数.
   value <R> 指定値.
4. 備考

なし.

5. 関連項目
   - 関連ルーチン (gnmlib)

10.2.9 DclSetGoodNumList

1. 機能
   「きりのよい数」のリストを設定する.
2. 書式
   ```
   call DclSetGoodNumList(list)
   ```
3. 引数
list	<R(:)>	きりのよい数を昇順に納めた実数型配列. list(1)=1.0, list(NB)=10.0 となるように指定する.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (gnmlib)

10.2.10 DclGetGoodNumList

1. 機能
   現在設定されている「きりのよい数」のリストを参照する.
2. 書式
   ```
   call DclGetGoodNumList(list)
   ```
3. 引数
list	<R(:)>	きりのよい数を昇順に納めた実数型配列.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (gnmlib)

10.2.11 DclSaveGoodNumList

1. 機能
   現在設定されている「きりのよい数」のリストを保存する.
2. 書式
   ```
   call DclSaveGoodNumList()
   ```
3. 引数なし.
4. 備考
   なし.

5. 関連項目
    - 関連ルーチン (gnmlib)

10.2.12 DclRestoreGoodNumList

1. 機能

    保存されたリストで「きりのよい数」を再設定する.
2. 書式

```
call DclRestoreGoodNumList()
```
3. 引数なし.
4. 備考

    なし.
5. 関連項目
    - 関連ルーチン (gnmlib)

第 11 章

実数に近い整数

11.1 ルーチンリスト

DclIntLT	指定値より小さい
(IRLT)	最大の整数を求める.
DclIntGT	指定値より大きい
(IRGT)	最小の整数を求める.
DclIntLE	指定値以下の
(IRLE)	最大の整数を求める.
DclIntGE	指定値以上の
(IRGE)	最小の整数を求める.

* 括弧の中は, 対応する f77 インターフェイス名.

11.2 各ルーチンの説明

11.2.1 DclIntLT

1. 機能

   指定値より小さい最大の整数を求める.

2. 書式

   ```
   result=DclIntLT(value)
   ```

3. 引数

戻り値	<I>	求める整数値.
value	<R>	指定値.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (intlib)

11.2.2 DclIntGE

1. 機能
   指定値より大きい最小の整数を求める.
2. 書式
   ```
   result=DclIntGE(value)
   ```
3. 引数
   戻り値 <I> 求める整数値.
   value <R> 指定値.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (intlib)

11.2.3 DclIntLE

1. 機能
   指定値以下の最大の整数を求める.
2. 書式
   ```
   result=DclIntLE(value)
   ```
3. 引数
   戻り値 <I> 求める整数値.
   value <R> 指定値.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (intlib)

11.2.4 DclIntGE

1. 機能
   指定値以上の最小の整数を求める.
2. 書式
   ```
   result=DclIntGE(value)
   ```
3. 引数
   戻り値 <I> 求める整数値.
   value <R> 指定値.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (intlib)

第 12 章

欠損値処理つき統計量

12.1 ルーチンリスト

DclGetAVE	平均を求める.
(RAVE)	
DclGetVAR	分散を求める.
(RVAR)	
DclGetSTD	標準偏差を求める.
(RSTD)	
DclGetRMS	root mean square を求める.
(RRMS)	
DclGetAMP	大きさ $((\sum rx^2)^{1/2})$ を求める.
(RAMP)	

* 括弧の中は, 対応する f77 インターフェイス名.

12.2 各ルーチンの説明

12.2.1 DclGetAVE

1. 機能

   平均を求める.

2. 書式

   ```
   result=DclGetAVE(x)
   ```

3. 引数

戻り値	<R>	平均値.
x	<R(:)>	入力データ.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (rfalib)

12.2.2 DclGetVAR

1. 機能

   分散を求める.

2. 書式

   ```
   result=DclGetVAR(x)
   ```

3. 引数

戻り値	<R>	分散.
x	<R(:)>	入力データ.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (rfalib)

12.2.3 DclGetSTD

1. 機能

   標準偏差を求める.

2. 書式

   ```
   result=DclGetSTD(x)
   ```

3. 引数

戻り値	<R>	標準偏差.
x	<R(:)>	入力データ.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (rfalib)

12.2.4 DclGetRMS

1. 機能

   root mean square を求める.

2. 書式

   ```
   result=DclGetRMS(x)
   ```

3. 引数

戻り値	<R>	二乗平均.
x	<R(:)>	入力データ.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (rfalib)

12.2.5 DclGetAMP

1. 機能

   大きさ ($\left(\sum x^2\right)^{1/2}$) を求める.

2. 書式

   ```
   result=DclGetAMP(x)
   ```

3. 引数

戻り値	<R>	$\left(\sum x^2\right)^{1/2}$.
x	<R(:)>	入力データ.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (rfalib)

第 13 章

補間

13.1 ルーチンリスト

DclInterpolate　実数または複素数型配列の補間をする.

* 括弧の中は, 対応する f77 インターフェイス名.

13.2 各ルーチンの説明

13.2.1 DclInterpolate

1. 機能

   実数または複素数型配列の RMISS の部分を線形補間する.

2. 書式

   ```
   result=DclInterpolate(x)
   ```

3. 引数

   戻り値　<R(:)>　補間されたデータ.

   x　　<R(:)>　欠損値を含む入力データ.

4. 備考

   - 処理するデータ列の始めまたは終りの部分に 'RMISS' であらわされる実数値がある場合, その部分の処理はおこなわれない.

5. 関連項目

   - 関連ルーチン (itrlib)

第 14 章

移動平均

14.1 ルーチンリスト

DclRunningMean (VRRNM)	移動平均を計算する.

* 括弧の中は, 対応する f77 インターフェイス名.

14.2 各ルーチンの説明

14.2.1 DclRunningMean

1. 機能

   移動平均を計算する.

2. 書式

   ```
   result=DclRunningMean(x, length)
   ```

3. 引数

戻り値	<R(:)>	移動平均されたデータ. 長さは x と同じ. 両端には欠損値が代入される.
x	<R(:)>	入力データ.
length	<I>	移動平均をとるデータ長. 1 以上で size(x) 以下, かつ奇数でなければならない.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (rnmlib)

第 15 章

座標変換

15.1 ルーチンリスト

DclConv2D	2 次元直角座標に変換する.
(CT2PC,CT2EC,CT2BC,CT2HC)	
DclConvPolar	2 次元極座標に変換する.
(CT2CP)	
DclConvHyperbolic	直角双曲線座標に変換する.
(CT2CH)	
DclConv3D	3 次元直角座標に変換する.
(CT3SC)	
DclConvSpherical	3 次元球面座標に変換する.
(CT3CS)	
DclRotate2D	2 次元直角座標を回転する.
(CR2C)	
DclRotate3D	3 次元直角座標を回転する.
(CR3C)	
DclRotateSpherical	3 次元球面座標を回転する.
(CR3S)	

* 括弧の中は, 対応する f77 インターフェイス名.

15.2 各ルーチンの説明

15.2.1 DclConv2D

1. 機能

   2 次元直角座標に変換する.

2. 書式

   ```
   result=DclConv2D(point)
   ```

3. 引数

戻り値	<cartesian>	2 次元直角座標値.
point	<polar, elliptic, bipolar, hyperbolic>	それぞれの型の座標値.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (ctrlib)

15.2.2 DclConv3D

1. 機能

   3 次元直角座標に変換する.

2. 書式

```
result=DclConv3D(point)
```

3. 引数

戻り値	<cartesian 3D>	3 次元直角座標値.
point	<spherical>	球座標値.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (ctrlib)

15.2.3 DclConvHyperbolic

1. 機能

   双曲線座標に変換する.

2. 書式

```
result=DclConvHyperbolic(point)
```

3. 引数

戻り値	<hyperbolic>	双曲線座標値.
point	<cartesian>	2 次元直角座標値.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (ctrlib)

15.2.4 DclConvPolar

1. 機能

   2 次元極座標に変換する.

2. 書式

```
result=DclConvPolar(point)
```

3. 引数

戻り値	<polar>	2 次元極座標値.
point	<cartesian>	2 次元直角座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (ctrlib)

15.2.5 DclConvSpherical

1. 機能

3 次元球面座標に変換する.

2. 書式

```
result=DclConvSpherical(point)
```

3. 引数

戻り値	<spherical>	3 次元球面座標値.
point	<cartesian 3D>	3 次元直角座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (ctrlib)

15.2.6 DclRotate2D

1. 機能

2 次元直角座標を回転する.

2. 書式

```
result=DclRotate2D(theta, point)
```

3. 引数

戻り値	<cartesian>	回転した座標値.
theta	<R>	回転角.
point	<cartesian>	もとの直角座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (ctrlib)

15.2.7 DclRotate3D

1. 機能

3 次元直角座標を回転する.

2. 書式

```
result=DclRotate3D(theta, phi, psi, point)
```

3. 引数

戻り値	<cartesian>	回転した 3 次元直角座標値.
theta, phi, psi	<R>	オイラーの回転角.
point	<cartesian3D>	もとの 3 次元直角座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (ctrlib)

15.2.8 DclRotateSpherical

1. 機能

3 次元球面座標を回転する.

2. 書式

```
result=DclRotateSpherical(theta, phi, psi, point)
```

3. 引数

戻り値	<cartesian>	回転した 3 次元球面座標値.
theta, phi, psi	<R>	オイラーの回転角.
point	<spherical>	もとの 3 次元球面座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (ctrlib)

第 16 章

地図投影

16.1 ルーチンリスト

■正距円筒図法

DclCylindrical_F　正変換
DclCylindrical_B　逆変換

■メルカトール図法

DclMercator_F　正変換
DclMercator_B　逆変換

■モルワイデ図法

DclMollweide_F　正変換
DclMollweide_B　逆変換

■擬似モルワイデ図法

DclMollweideLike_F　正変換
DclMollweideLike_B　逆変換

■ハンメル図法

DclHammer_F　正変換
DclHammer_B　逆変換

■エッケルト第 6 図法

DclEckert6_F　正変換
DclEckert6_B　逆変換

■北田楕円図法

DclKitada_F　正変換
DclKitada_B　逆変換

■円錐図法

DclConical_F 正変換
DclConical_B 逆変換
DclSetConical 標準緯線の指定

■ランベルト正積円錐図法

DclConicalA_F 正変換
DclConicalA_B 逆変換
DclSetConicalA 標準緯線の指定

■ランベルト正角円錐図法

DclConicalC_F 正変換
DclConicalC_B 逆変換
DclSetConicalC 標準緯線の指定

■ボンヌ図法

DclBonnes_F 正変換
DclBonnes_B 逆変換
DclSetBonnes 標準緯線の指定

■正射図法

DclOrthographic_F 正変換
DclOrthographic_B 逆変換
DclSetOrthographic 軌道半径の設定

■ポーラーステレオ図法

DclPolarStereo_F 正変換
DclPolarStereo_B 逆変換

■正距方位図法

DclAzimuthal_F 正変換
DclAzimuthal_B 逆変換

■ランベルト正積方位図法

DclAzimuthalA_F 正変換
DclAzimuthalA_B 逆変換

* 括弧の中は, 対応する f77 インターフェイス名.

16.2 各ルーチンの説明

16.2.1 DclCylindrical_F

1. 機能

   正距円筒図法　正変換.

2. 書式

   ```
   result=DclCylindrical_F(point)
   ```

3. 引数

戻り値	<cartesian>	直角座標値.
point	<map>	緯度経度座標値.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (maplib)

16.2.2 DclCylindrical_B

1. 機能

   正距円筒図法　逆変換.

2. 書式

   ```
   result=DclCylindrical_B(point)
   ```

3. 引数

戻り値	<map>	緯度経度座標値.
point	<cartesian>	直角座標値.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (maplib)

16.2.3 DclMercator_F

1. 機能

   メルカトール図法　正変換.

2. 書式

   ```
   result=DclMercator_F(point)
   ```

3. 引数

戻り値	<cartesian>	直角座標値.
point	<map>	緯度経度座標値.

4. 備考

   なし.

5. 関連項目
   - 関連ルーチン (maplib)

16.2.4 DclMercator_B

1. 機能
   メルカトール図法　逆変換.
2. 書式
   ```
   result=DclMercator_B(point)
   ```
3. 引数
   戻り値　<map>　緯度経度座標値.
   point　<cartesian>　直角座標値.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (maplib)

16.2.5 DclMollweide_F

1. 機能
   モルワイデ図法　正変換.
2. 書式
   ```
   result=DclMollweide_F(point)
   ```
3. 引数
   戻り値　<cartesian>　直角座標値.
   point　<map>　緯度経度座標値.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (maplib)

16.2.6 DclMollweide_B

1. 機能
   モルワイデ図法　逆変換.
2. 書式
   ```
   result=DclMollweide_B(point)
   ```
3. 引数
   戻り値　<map>　緯度経度座標値.
   point　<cartesian>　直角座標値.
4. 備考

なし.

5. 関連項目
   - 関連ルーチン (maplib)

16.2.7 DclMollweideLike_F

1. 機能
   モルワイデ図法もどき　正変換.
2. 書式
   ```
   result=DclMollweideLike_F(point)
   ```
3. 引数
   戻り値　<cartesian>　直角座標値.
   point　<map>　緯度経度座標値.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (maplib)

16.2.8 DclMollweideLike_B

1. 機能
   モルワイデ図法もどき　逆変換.
2. 書式
   ```
   result=DclMollweideLike_B(point)
   ```
3. 引数
   戻り値　<map>　緯度経度座標値.
   point　<cartesian>　直角座標値.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (maplib)

16.2.9 DclHammer_F

1. 機能
   ハンメル図法　正変換.
2. 書式
   ```
   result=DclHammer_F(point)
   ```
3. 引数
   戻り値　<cartesian>　直角座標値.
   point　<map>　緯度経度座標値.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (maplib)

16.2.10 DclHammer_B

1. 機能

   ハンメル図法　逆変換.

2. 書式

   ```
   result=DclHammer_B(point)
   ```

3. 引数

戻り値	<map>	緯度経度座標値.
point	<cartesian>	直角座標値.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (maplib)

16.2.11 DclEckert6_F

1. 機能

   エッケルト第 6 図法　正変換.

2. 書式

   ```
   result=DclEckert6_F(point)
   ```

3. 引数

戻り値	<cartesian>	直角座標値.
point	<map>	緯度経度座標値.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (maplib)

16.2.12 DclEckert6_B

1. 機能

   エッケルト第 6 図法　逆変換.

2. 書式

   ```
   result=DclEckert6_B(point)
   ```

3. 引数

戻り値 <map> 緯度経度座標値.
point <cartesian> 直角座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.13 DclKitada_F

1. 機能

北田楕円図法　正変換.

2. 書式

```
result=DclKitada_F(point)
```

3. 引数

戻り値 <cartesian> 直角座標値.
point <map> 緯度経度座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.14 DclKitada_B

1. 機能

北田楕円図法　逆変換.

2. 書式

```
result=DclKitada_B(point)
```

3. 引数

戻り値 <map> 緯度経度座標値.
point <cartesian> 直角座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.15 DclConical_F

1. 機能

円錐図法　正変換.

2. 書式

```
result=DclConical_F(point)
```

3. 引数

戻り値	`<cartesian>`	直角座標値.
`point`	`<map>`	緯度経度座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.16 DclConical_B

1. 機能

円錐図法　逆変換.

2. 書式

```
result=DclConical_B(point)
```

3. 引数

戻り値	`<map>`	緯度経度座標値.
`point`	`<cartesian>`	直角座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.17 DclSetConical

1. 機能

円錐図法の標準緯線を指定する.

2. 書式

```
call DclSetConical(lat)
```

3. 引数

`lat`	`<R>`	標準緯線.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.18 DclConicalA_F

1. 機能

ランベルト正積円錐図法　正変換.

2. 書式

```
result=DclConicalA_F(point)
```

3. 引数

戻り値	`<cartesian>`	直角座標値.
`point`	`<map>`	緯度経度座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.19 DclConicalA_B

1. 機能

ランベルト正積円錐図法　逆変換.

2. 書式

```
result=DclConicalA_B(point)
```

3. 引数

戻り値	`<map>`	緯度経度座標値.
`point`	`<cartesian>`	直角座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.20 DclSetConicalA

1. 機能

ランベルト正積円錐図法の標準緯線を指定する.

2. 書式

```
call DclSetConicalA(lat)
```

3. 引数

`lat`	`<R>`	標準緯線.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.21 DclConicalC_F

1. 機能

ランベルト正角円錐図法　正変換.

2. 書式

```
result=DclConicalC_F(point)
```

3. 引数

戻り値	`<cartesian>`	直角座標値.
`point`	`<map>`	緯度経度座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.22 DclConicalC_B

1. 機能

ランベルト正角円錐図法　逆変換.

2. 書式

```
result=DclConicalC_B(point)
```

3. 引数

戻り値	`<map>`	緯度経度座標値.
`point`	`<cartesian>`	直角座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.23 DclSetConicalC

1. 機能

ランベルト正角円錐図法の標準緯線を指定する.

2. 書式

```
call DclSetConicalC(lat1, lat2)
```

3. 引数

`lat1, lat2`	`<R>`	標準緯線.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.24 DclBonnes_F

1. 機能

ボンヌ図法　正変換.

2. 書式

```
result=DclBonnes_F(point)
```

3. 引数

戻り値	`<cartesian>`	直角座標値.
`point`	`<map>`	緯度経度座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.25 DclBonnes_B

1. 機能

ボンヌ図法　逆変換.

2. 書式

```
result=DclBonnes_B(point)
```

3. 引数

戻り値	`<map>`	緯度経度座標値.
`point`	`<cartesian>`	直角座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.26 DclSetBonnes

1. 機能

ボンヌ図法の標準緯線を指定する.

2. 書式

```
call DclSetBonnes(lat)
```

3. 引数

`lat`	`<R>`	標準緯線.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.27 DclOrthographic_F

1. 機能

正射図法　正変換.

2. 書式

```
result=DclOrthographic_F(point)
```

3. 引数

戻り値	`<cartesian>`	直角座標値.
`point`	`<map>`	緯度経度座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.28 DclOrthographic_B

1. 機能

正射図法　逆変換.

2. 書式

```
result=DclOrthographic_B(point)
```

3. 引数

戻り値	`<map>`	緯度経度座標値.
`point`	`<cartesian>`	直角座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.29 DclSetOrthographic

1. 機能

正射図法の軌道半径を指定する.

2. 書式

```
call DclSetOrthographic(rsat)
```

3. 引数

`rsat`	`<R>`	軌道半径.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.30 DclPolarStereo_F

1. 機能

ポーラーステレオ図法　正変換.

2. 書式

```
result=DclPolarStereo_F(point)
```

3. 引数

戻り値	`<cartesian>`	直角座標値.
`point`	`<map>`	緯度経度座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.31 DclPolarStereo_B

1. 機能

ポーラーステレオ図法　逆変換.

2. 書式

```
result=DclPolarStereo_B(point)
```

3. 引数

戻り値	`<map>`	緯度経度座標値.
`point`	`<cartesian>`	直角座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.32 DclAzimuthal_F

1. 機能

正距方位図法　正変換.

2. 書式

```
result=DclAzimuthal_F(point)
```

3. 引数

戻り値	`<cartesian>`	直角座標値.
`point`	`<map>`	緯度経度座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.33 DclAzimuthal_B

1. 機能

正距方位図法　逆変換.

2. 書式

```
result=DclAzimuthal_B(point)
```

3. 引数

戻り値	<map>	緯度経度座標値.
point	<cartesian>	直角座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.34 DclAzimuthalA_F

1. 機能

ランベルト正積方位図法　正変換.

2. 書式

```
result=DclAzimuthalA_F(point)
```

3. 引数

戻り値	<cartesian>	直角座標値.
point	<map>	緯度経度座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

16.2.35 DclAzimuthalA_B

1. 機能

ランベルト正積方位図法　逆変換.

2. 書式

```
result=DclAzimuthalA_B(point)
```

3. 引数

戻り値	<map>	緯度経度座標値.
point	<cartesian>	直角座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (maplib)

第 17 章

フーリエ変換

17.1 概要

任意の長さのデータについて高速フーリエ変換をおこなうサブルーチンパッケージ. NCAR の数値計算ライブラリより移植した.

サブルーチンの説明の中の「定義」の節では, 入出力パラメータの数学的解説に関して次のような表記法をとる: 処理する配列 X (長さ N) の第 i 番目 ($i = 1, \ldots, \mathtt{N}$) の配列要素について, 入力時の値を x_i (小文字), 出力時の値を X_i (大文字) と書く.

以下の 7 つのサブルーチン群の中で初期化をおこなうサブルーチン (サブルーチン名が I で終わる) は, そのサブルーチン群に属する変換ルーチンを用いる際, かならず最初に 1 回呼ばなければならない. ただしそれ以後は, 異なるデータ数を指定するときに限って初期化ルーチンを呼べばよい. なお, 初期化ルーチンが用いる作業領域は, 同じサブルーチン群に属する変換ルーチンを用いている間変更してはならない. (この作業領域には, 因数と三角関数表が格納されている).

17.2 ルーチンリスト

■周期実数値データのフーリエ変換

DclInitRealFFT	初期化.
DclDeallocRealFFT	作業領域の開放.
DclRealFFT_F	順変換.
DclRealFFT_B	逆変換.

■周期実数値データのフーリエ変換 (簡易版)

DclInitEasyFFT	初期化.
DclDeallocEasyFFT	作業領域の開放.
DclEasyFFT_F	順変換.
DclEasyFFT_B	逆変換.

■奇の周期データの SINE 変換

DclInitSinFFT	初期化.
DclDeallocSinFFT	作業領域の開放.
DclSinFFT	変換 (正逆共通).

■偶の周期データの COSINE 変換

DclInitCosFFT	初期化.
DclDeallocCosFFT	作業領域の開放.
DclCosFFT	変換 (正逆共通).

■奇数波数成分のみの SINE 変換

DclInitSinQFT	初期化.
DclDeallocSinQFT	作業領域の開放.
DclSinQFT_F	順変換.
DclSinQFT_B	逆変換.

■偶数波数成分のみの COSINE 変換

DclInitCosQFT	初期化.
DclDeallocCosQFT	作業領域の開放.
DclCosQFT_F	順変換.
DclCosQFT_B	逆変換.

■周期複素数値データのフーリエ変換

DclInitComplexFFT	初期化.
DclDeallocComplexFFT	作業領域の開放.
DclComplexFFT_F	順変換.
DclComplexFFT_B	逆変換.

* 括弧の中は, 対応する f77 インターフェイス名.

17.3 各ルーチンの説明

17.3.1 DclInitRealFFT

1. 機能

   周期実数値データのフーリエ順変換の初期化をする.

2. 書式

   ```
   call DclInitRealFFT(n, [index])
   ```

3. 引数

n	<I>	変換するデータの長さ (個数).
index	<I>	作業領域番号. 省略値は 1.

定義 N が偶数のとき $N' = N/2 - 1$, N が奇数のとき $N' = (N-1)/2$ とおく.

順変換は次のように定義される.

$$R_1 = \sum_{i=1}^{N} r_i,$$

$$R_{2k} = \sum_{i=1}^{N} r_i \cos\frac{2\pi(i-1)k}{N}, \quad R_{2k+1} = -\sum_{i=1}^{N} r_i \sin\frac{2\pi(i-1)k}{N} \quad (k = 1, \ldots, N').$$

ただし N が偶数のとき,

$$R_N = \sum_{i=1}^{N} (-1)^{i-1} r_i.$$

逆変換は次のように定義される.

N が偶数のとき,

$$R_i = r_1 + (-1)^{i-1} r_N + 2\sum_{k=1}^{N'} (r_{2k} \cos\frac{2\pi(i-1)k}{N} - r_{2k+1} \sin\frac{2\pi(i-1)k}{N}) \quad (i = 1, \ldots, N).$$

N が奇数のとき,

$$R_i = r_1 + 2\sum_{k=1}^{N'} (r_{2k} \cos\frac{2\pi(i-1)k}{N} - r_{2k+1} \sin\frac{2\pi(i-1)k}{N}) \quad (i = 1, \ldots, N).$$

4. 備考
   - この変換では正規化されない. つまり `DclRealFFT_F, DclRealFFT_B` を続けて呼ぶと, もとの N 倍の値が返される.
5. 関連項目
   - 関連ルーチン (fftlib)

17.3.2 DclDeallocRealFFT

1. 機能

   周期実数値データのフーリエ順変換の作業領域を開放する.
2. 書式

   ```
   call DclDeallocRealFFT([index])
   ```
3. 引数

   `index` `<I>` 作業領域番号.
4. 備考

   なし.
5. 関連項目
   - 関連ルーチン (fftlib)

17.3.3 DclRealFFT_F

1. 機能

周期実数値データのフーリエ順変換をする.

2. 書式

```
result=DclRealFFT_F(x, [index])
```

3. 引数

戻り値	<R(:)>	変換されたデータ.
x	<R(:)>	変換するデータ.
index	<I>	作業領域番号.

4. 備考

- この変換では正規化されない. つまり DclRealFFT_F, DclRealFFT_B を続けて呼ぶと, もとの N 倍の値が返される.

5. 関連項目

- 関連ルーチン (fftlib)

17.3.4 DclRealFFT_B

1. 機能

周期実数値データのフーリエ逆変換をする.

2. 書式

```
result=DclRealFFT_B(x, [index])
```

3. 引数

戻り値	<R(:)>	変換されたデータ.
x	<R(:)>	変換するデータ.
index	<I>	作業領域番号.

4. 備考

- この変換では正規化されない. つまり DclRealFFT_F, DclRealFFT_B を続けて呼ぶと, もとの N 倍の値が返される.

5. 関連項目

- 関連ルーチン (fftlib)

17.3.5 DclInitEasyFFT

1. 機能

周期実数値データのフーリエ変換 (簡易版) の初期化をする.

2. 書式

```
call DclInitEasyFFT(n, [index])
```

3. 引数

n	<I>	変換するデータの長さ (個数).
index	<I>	作業領域番号. 省略値は 1.

定義 **順変換**は次のように定義される. (N が偶数のとき $N' = N/2-1$, N が奇数のとき $N' = (N-1)/2$

とおく.)

$$A_0 = \frac{1}{N}\sum_{i=1}^{N} r_i,$$

$$A_k = \frac{2}{N}\sum_{i=1}^{N} r_i \cos\frac{2\pi(i-1)k}{N}, \quad B_k = \frac{2}{N}\sum_{i=1}^{N} r_i \sin\frac{2\pi(i-1)k}{N} \quad (k = 1, \ldots, N').$$

ただし N が偶数のとき,

$$A_{N/2} = \frac{1}{N}\sum_{i=1}^{N}(-1)^{i-1} r_i, \quad B_{N/2} = 0.$$

逆変換は次のように定義される. (N が偶数のとき $N' = N/2$, N が奇数のとき $N' = (N-1)/2$ とおく.)

$$R_i = a_0 + \sum_{k=1}^{N'}(a_k \cos\frac{2\pi(i-1)k}{N} + b_k \sin\frac{2\pi(i-1)k}{N}) \quad (i = 1, \ldots, N).$$

4. 備考
   - この変換では正規化される. つまり `DclEasyFFT_F, DclEasyFFT_B` を続けて呼ぶと, もとの値が返される.
5. 関連項目
   - 関連ルーチン (fftlib)

17.3.6 DclDeallocEasyFFT

1. 機能

   周期実数値データのフーリエ変換 (簡易版) の作業領域を開放する.
2. 書式

   ```
   call DclDeallocEasyFFT([index])
   ```
3. 引数

   `index` `<I>` 作業領域番号.
4. 備考

   なし.
5. 関連項目
   - 関連ルーチン (fftlib)

17.3.7 DclEasyFFT_F

1. 機能

   周期実数値データのフーリエ順変換 (簡易版) をする.
2. 書式

   ```
   call DclEasyFFT_F(x, a0, a, b, [index])
   ```
3. 引数

x	<R(:)>	入力データ.
a0	<R>	cos 第 0 成分.
a	<R(:)>	cos 成分.
b	<R(:)>	sin 成分.
index	<I>	作業領域番号.

4. 備考
   - この変換では正規化される. つまり DclEasyFFT_F, DclEasyFFT_B を続けて呼ぶと, もとの値が返される.
5. 関連項目
   - 関連ルーチン (fftlib)

17.3.8 DclEasyFFT_B

1. 機能
   周期実数値データのフーリエ逆変換 (簡易版) をする.
2. 書式
```
call DclEasyFFT_B(x, a0, a, b, [index])
```
3. 引数

x	<R(:)>	入力データ.
a0	<R>	cos 第 0 成分.
a	<R(:)>	cos 成分.
b	<R(:)>	sin 成分.
index	<I>	作業領域番号.

4. 備考
   - この変換では正規化される. つまり DclEasyFFT_F, DclEasyFFT_B を続けて呼ぶと, もとの値が返される.
5. 関連項目
   - 関連ルーチン (fftlib)

17.3.9 DclInitSinFFT

1. 機能
   奇の周期データの SINE 変換の初期化をする.
2. 書式
```
call DclInitSinFFT(n, [index])
```
3. 引数

n	<I>	変換するデータの長さ (個数).
index	<I>	作業領域番号. 省略値は 1.

定義式 **順変換**は次のように定義される.

$$X_k = 2\sum_{i=1}^{N} x_i \sin\frac{\pi ik}{N+1}, \quad (k = 1, \ldots, N).$$

逆変換は順変換と同じである.

4. 備考
   - DclSinFFT は逆変換でもある. また, この変換では正規化されない. つまり DclSinFFT を 2 回続けて呼ぶと, もとの $2(N+1)$ 倍の値が返される.
5. 関連項目
   - 関連ルーチン (fftlib)

17.3.10 DclDeallocSinFFT

1. 機能

   奇の周期データの SINE 変換の作業領域を開放する.
2. 書式

   ```
   call DclDeallocSinFFT([index])
   ```
3. 引数

index	<I>	作業領域番号.
4. 備考

   なし.
5. 関連項目
   - 関連ルーチン (fftlib)

17.3.11 DclSinFFT

1. 機能

   奇の周期データの SINE 変換をする.
2. 書式

   ```
   result=DclSinFFT(x, [index])
   ```
3. 引数

戻り値	<R(:)>	変換されたデータ.
x	<R(:)>	変換するデータ.
index	<I>	作業領域番号.
4. 備考
   - DclSinFFT は逆変換でもある. また, この変換では正規化されない. つまり DclSinFFT を 2 回続けて呼ぶと, もとの $2(N+1)$ 倍の値が返される.
5. 関連項目
   - 関連ルーチン (fftlib)

17.3.12 DclInitCosFFT

1. 機能

   偶の周期データの COSINE 変換の初期化をする.

2. 書式

   ```
   call DclInitCosFFT(n, [index])
   ```

3. 引数

x	<I>	変換するデータの長さ (個数).
index	<I>	作業領域番号. 省略値は 1.

定義　**順変換**は次のように定義される.

$$X_k = x_1 + (-1)^{k-1} x_N + 2\sum_{i=2}^{N} x_i \cos\frac{\pi(i-1)(k-1)}{N-1}, \quad (k = 1, \ldots, N).$$

逆変換は順変換と同じである.

4. 備考

   - DclCosFFT は逆変換でもある. また, この変換では正規化されない. つまり DclCosFFT を 2 回続けて呼ぶと, もとの $2(N-1)$ 倍の値が返される.

5. 関連項目

   - 関連ルーチン (fftlib)

17.3.13 DclDeallocCosFFT

1. 機能

   偶の周期データの COSINE 変換の作業領域を開放する.

2. 書式

   ```
   call DclDeallocCosFFT([index])
   ```

3. 引数

index	<I>	作業領域番号.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (fftlib)

17.3.14 DclCosFFT

1. 機能

   偶の周期データの COSINE 変換をする.

2. 書式

   ```
   result=DclCosFFT(x, [index])
   ```

3. 引数

戻り値 <R(:)> 変換されたデータ.
x <R(:)> 変換するデータ.
index <I> 作業領域番号.

4. 備考
   - DclCosFFT は逆変換でもある. また, この変換では正規化されない. つまり DclCosFFT を 2 回続けて呼ぶと, もとの $2(N-1)$ 倍の値が返される.
5. 関連項目
   - 関連ルーチン (fftlib)

17.3.15 DclInitSinQFT

1. 機能
   奇数波数成分のみの SIN 順変換の初期化をする.
2. 書式
   ```
   call DclInitSinQFT(n, [index])
   ```
3. 引数
   n <I> 変換するデータの長さ (個数).
   index <I> 作業領域番号. 省略値は 1.

定義 **順変換**は次のように定義される.

$$X_k = (-1)^{k-1}x_N + 2\sum_{i=1}^{N-1} r_i \sin\frac{\pi(2k-1)i}{2N} \quad (k = 1, \ldots, N).$$

逆変換は次のように定義される.

$$X_i = 4\sum_{k=1}^{N} r_k \sin\frac{\pi(2k-1)i}{2N} \quad (i = 1, \ldots, N).$$

4. 備考
   - この変換では正規化されない. つまり DclSinQFT_F, DclSinQFT_B を続けて呼ぶと, もとの $4N$ 倍の値が返される.
5. 関連項目
   - 関連ルーチン (fftlib)

17.3.16 DclDeallocSinQFT

1. 機能
   奇数波数成分のみの SINE 変換の作業領域を開放する.
2. 書式
   ```
   call DclDeallocSinQFT([index])
   ```
3. 引数
   index <I> 作業領域番号.
4. 備考

なし.

5. 関連項目
   - 関連ルーチン (fftlib)

17.3.17 DclSinQFT_F

1. 機能

   奇数波数成分のみの SIN 順変換をおこなう.

2. 書式

```
result=DclSinQFT_F(x, [index])
```

3. 引数

戻り値	<R(:)>	変換されたデータ.
x	<R(:)>	変換するデータ.
index	<I>	作業領域番号.

4. 備考
   - この変換では正規化されない. つまり DclSinQFT_F, DclSinQFT_B を続けて呼ぶと, もとの $4N$ 倍の値が返される.

5. 関連項目
   - 関連ルーチン (fftlib)

17.3.18 DclSinQFT_B

1. 機能

   奇数波数成分のみの SIN 逆変換をおこなう.

2. 書式

```
result=DclSinQFT_B(x, [index])
```

3. 引数

戻り値	<R(:)>	変換されたデータ.
x	<R(:)>	変換するデータ.
index	<I>	作業領域番号.

4. 備考
   - この変換では正規化されない. つまり DclSinQFT_F, DclSinQFT_B を続けて呼ぶと, もとの $4N$ 倍の値が返される.

5. 関連項目
   - 関連ルーチン (fftlib)

17.3.19 DclInitCosQFT

1. 機能

   偶数波数成分のみの COSINE 順変換をおこなう.

2. 書式

```
call DclInitCosQFT(n, [index])
```

3. 引数

n	<I>	変換するデータの長さ (個数).
index	<I>	作業領域番号. 省略値は 1.

定義 **順変換**は次のように定義される.

$$X_k = x_1 + 2\sum_{i=2}^{N} r_i \cos\frac{\pi(2k-1)(i-1)}{2N} \quad (k = 1, \ldots, N).$$

逆変換は次のように定義される.

$$X_i = 4\sum_{k=1}^{N} r_k \cos\frac{\pi(2k-1)(i-1)}{2N} \quad (i = 1, \ldots, N).$$

4. 備考

- この変換では正規化されない. つまり DclCosQFT_F, DclCosQFT_B を続けて呼ぶと, もとの $4N$ 倍の値が返される.

5. 関連項目

- 関連ルーチン (fftlib)

17.3.20 DclDeallocCosQFT

1. 機能

偶数波数成分のみの COSIN 変換の作業領域を開放する.

2. 書式

```
call DclDeallocCosQFT([index])
```

3. 引数

index	<I>	作業領域番号.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (fftlib)

17.3.21 DclCosQFT_F

1. 機能

偶数波数成分のみの COSINE 順変換をおこなう.

2. 書式

```
result=DclCosQFT_F(x, [index])
```

3. 引数

戻り値	<R(:)>	変換されたデータ.
x	<R(:)>	変換するデータ.
index	<I>	作業領域番号.

4. 備考
   - この変換では正規化されない. つまり DclCosQFT_F, DclCosQFT_B を続けて呼ぶと, もとの $4N$ 倍の値が返される.
5. 関連項目
   - 関連ルーチン (fftlib)

17.3.22 DclCosQFT_B

1. 機能
   偶数波数成分のみの COSINE 逆変換をおこなう.
2. 書式
   ```
   result=DclCosQFT_B(x, [index])
   ```
3. 引数
   戻り値 <R(:)> 変換されたデータ.
   x <R(:)> 変換するデータ.
   index <I> 作業領域番号.
4. 備考
   - この変換では正規化されない. つまり DclCosQFT_F, DclCosQFT_B を続けて呼ぶと, もとの $4N$ 倍の値が返される.
5. 関連項目
   - 関連ルーチン (fftlib)

17.3.23 DclInitComplexFFT

1. 機能
   周期複素数データのフーリエ順変換をおこなう.
2. 書式
   ```
   call DclInitComplexFFT(n, [index])
   ```
3. 引数
   n <I> 変換するデータの長さ (個数).
   index <I> 作業領域番号. 省略値は 1.

定義 以下では $i = \sqrt{-1}$ とする.

順変換は次のように定義される.

$$C_k = \sum_{j=1}^{N} c_j \exp(-i\frac{2\pi(j-1)(k-1)}{N}) \quad (k = 1, \ldots, N).$$

逆変換は次のように定義される.

$$C_j = \sum_{k=1}^{N} c_k \exp(i\frac{2\pi(j-1)(k-1)}{N}) \quad (j = 1, \ldots, N).$$

4. 備考

- この変換では正規化されない. つまり DclComplexFFT_F, DclComplexFFT_B を続けて呼ぶと, もとの N 倍の値が返される.

5. 関連項目
   - 関連ルーチン (fftlib)

17.3.24 DclDeallocComplexFFT

1. 機能

   周期複素数データのフーリエ変換の作業領域を開放する.

2. 書式

```
call DclDeallocComplexFFT([index])
```

3. 引数

   index <I> 作業領域番号.

4. 備考

   なし.

5. 関連項目
   - 関連ルーチン (fftlib)

17.3.25 DclComplexFFT_F

1. 機能

   周期複素数データのフーリエ順変換をおこなう.

2. 書式

```
result=DclComplexFFT_F(x, [index])
```

3. 引数

   戻り値 <X(:)> 変換されたデータ.
   x <X(:)> 変換するデータ.
   index <I> 作業領域番号.

4. 備考
   - この変換では正規化されない. つまり DclComplexFFT_F, DclComplexFFT_B を続けて呼ぶと, もとの N 倍の値が返される.

5. 関連項目
   - 関連ルーチン (fftlib)

17.3.26 DclComplexFFT_B

1. 機能

   周期複素数データのフーリエ逆変換をおこなう.

2. 書式

```
result=DclComplexFFT_B(x, [index])
```

3. 引数

戻り値	`<X(:)>`	変換されたデータ.
`x`	`<X(:)>`	変換するデータ.
`index`	`<I>`	作業領域番号.

4. 備考
   - この変換では正規化されない. つまり `DclComplexFFT_F, DclComplexFFT_B` を続けて呼ぶと, もとの N 倍の値が返される.
5. 関連項目
   - 関連ルーチン (fftlib)

第 18 章

球面調和関数

18.1 概要

これは, スペクトル (球面調和関数) 変換を行なうサブルーチンパッケージであり, 球面調和関数展開の係数からグリッドデータ、およびその逆の変換を行なう. このパッケージは, データ解析を念頭において設計されており, 等間隔グリッドデータを扱えるという特長がある. また, スペクトルで与えられたデータの解析を念頭におき, 逆変換系のルーチンを充実させている. このパッケージの内部では FFTLIB のサブルーチンを用いている.

切断波数 M(三角切断) のスペクトル逆変換は, 以下のように表せる:

$$G(\lambda,\varphi)=\sum_{n=0}^{M}\sum_{m=-n}^{n}S_n^m P_n^m(\sin\varphi)e^{im\lambda}. \tag{18.1}$$

または, ルジャンドル逆変換:

$$W^m(\varphi)\equiv\sum_{n=|m|}^{M}S_n^m P_n^m(\sin\varphi) \tag{18.2}$$

を導入すると、

$$G(\lambda,\varphi)=\sum_{m=-M}^{M}W^m(\varphi)e^{im\lambda} \tag{18.3}$$

と, ルジャンドル逆変換とフーリエ逆変換の積として表される. ここに, λ: 経度, φ: 緯度である.

また, $P_n^m(\mu)$ は 2 に正規化されたルジャンドル陪関数で, 以下のように定義される:

$$P_n^m(\mu)\equiv\sqrt{(2n+1)\frac{(n-m)!}{(n+m)!}}\frac{1}{2^n n!}(1-\mu^2)^{m/2}\frac{d^{n+m}}{d\mu^{n+m}}(\mu^2-1)^n, \tag{18.4}$$

$$\int_{-1}^{1}\{P_n^m(\mu)\}^2 d\mu=2. \tag{18.5}$$

また, スペクトル逆変換は以下のように表せる:

$$S_n^m=\frac{1}{4\pi}\int_0^{2\pi}\int_{-\pi/2}^{\pi/2}G(\lambda,\varphi)P_n^m(\sin\varphi)e^{-im\lambda}\cos\varphi d\varphi d\lambda. \tag{18.6}$$

逆変換の場合と同様に, フーリエ正変換を,

$$W^m(\varphi) \equiv \frac{1}{2\pi}\int_0^{2\pi} G(\lambda,\varphi)e^{-im\lambda}d\lambda \tag{18.7}$$

と導入すると,

$$S_n^m = \frac{1}{2}\int_{-\pi/2}^{\pi/2} W^m(\varphi)P_n^m(\sin\varphi)\cos\varphi d\varphi \tag{18.8}$$

と, フーリエ正変換とルジャンドル正変換の積として表される.

$G(\lambda,\varphi)$ が実数であるとすると, S_n^m および $W^m(\varphi)$ は以下の関係を満たしている必要がある:

$$W^{-m}(\varphi) = \{W^m(\varphi)\}^* \tag{18.9}$$

$$S_n^{-m} = \{S_n^m\}^* \tag{18.10}$$

ここに, $\{\}^*$ は複素共役を表す. 従って, $W^m(\sin\varphi)$ および S_n^m は $m \geq 0$ の範囲だけを求めれば良い. さらに, 上の制約から, $W^0(\sin\varphi)$ および S_n^0 は実数である.

本ライブラリは, スペクトルデータ $(S_n^m) \to$ 等間隔緯度円上のウエーブデータ $(W^m(\varphi_j)) \to$ 等間隔格子点上のグリッドデータ $(G(\lambda_i,\varphi_j))$ の逆変換を (1∼ 3) 式に基づいて行うルーチン群, 等間隔格子点上のグリッドデータ $(G(\lambda_i,\varphi_j)) \to$ 等間隔緯度円上のウエーブデータ $(W^m(\varphi_j)) \to$ スペクトルデータ (S_n^m) の正変換を (6∼8) 式に基づいて行うルーチン群およびそして, その他の補助ルーチン群よりなっている.

ここに, 格子点の経度 λ_i, 緯度 φ_j は分割数 I, J によって以下のように定められるものとする:

$$\lambda_i = \frac{\pi i}{I}, \quad i = -I, -I+1, \cdots, 0, \cdots, I-1, I, \tag{18.11}$$

$$\varphi_j = \frac{\pi j}{2J}, \quad j = -J, -J+1, \cdots, 0, \cdots, J-1, J. \tag{18.12}$$

18.2 ルーチンリスト

DclInitSHT	初期化する.
(SHTINT)	
DclDeallocSHT	作業領域を解放する.
(----)	
DclGetSpectrumNumber	スペクトルデータの
(SHTNML)	格納位置を求める.
DclOperateLaplacian	スペクトルデータに対して
(SHTLAP)	ラプラシアンを演算する.
DclSpectrumToGrid	スペクトルデータから
(SHTS2G)	グリッドデータへ変換する.
DclGridToSpectrum	グリッドデータから
(SHTG2W)	スペクトルデータへ変換する.
DclSpectrumToGridForWave	スペクトルデータからグリッド
(SHTWGM,SHTSGM,SHTSWM)	データへ変換する.(波数指定)
DclSpectrumToGridForZonal	スペクトルデータからグリッド
(SHTWGZ,SHTSGZ,SHTSWZ)	データへ変換する.(帯状成分)
DclSpectrumToGridForLatitude	スペクトルデータからグリッド
(SHTSGJ,SHTSWJ,SHTGJ)	データへ変換する.(緯度円指定)
DclGetLegendreFunctions	ルジャンドル陪関数の計算.
(SHTFUN)	
DclLegendreTransform_F	ルジャンドル正変換.
(SHTLFW)	
DclLegendreTransform_B	ルジャンドル逆変換.
(SHTLBW)	

* 括弧の中は, 対応する f77 インターフェイス名.

18.3 各ルーチンの説明

18.3.1 DclInitSHT

1. 機能

   初期化する.

2. 書式

   ```
   call DclInitSHT(mm, jm, im, [idx])
   ```

3. 引数

mm	<I>	切断波数 (M).
jm	<I>	南北分割数の 1/2 (J).
im	<I>	東西分割数の 1/2 (I).
idx	<I>	作業領域番号. 省略値は 1.

4. 備考
   - jm ≧(mm+1)/2, im ≧ mm+1 でなければならない.
5. 関連項目
   - 関連ルーチン (shtlib)

18.3.2 DclDeallocSHT

1. 機能
   作業領域を開放する.
2. 書式
```
call DclDeallocSHT([idx])
```
3. 引数

idx	<I>	作業領域番号.

4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (shtlib)

18.3.3 DclGetSpectrumNumber

1. 機能
   スペクトルデータの格納位置を求める.
2. 書式
```
result=DclGetSpectrumNumber(n, m, [idx])
```
3. 引数

戻り値	<I>	データの格納位置.
n	<I>	全波数.
m	<I>	帯状波数.
idx	<I>	作業領域番号.

4. 備考
   - SHTLIB において, スペクトルデータ (S_n^m) は概要に述べた制限をもとに, 独立な $(M+1)^2$ 個の成分; $S_0^0, S_1^0, \cdots, S_M^0$, $\mathrm{Re}(S_1^1), \mathrm{Re}(S_2^1), \cdots, \mathrm{Re}(S_M^1)$, $\mathrm{Im}(S_1^1), \mathrm{Im}(S_2^1), \cdots, \mathrm{Im}(S_M^1)$, $\cdots, \mathrm{Re}(S_M^M), \mathrm{Im}(S_M^M)$ がこの順序で (長さ (MM+1)**2 の配列に) 格納されている. ここに, Re( ) は実数部を, Im( ) は虚数部を表す. このサブルーチンは切断波数 M, S_n^m の全波数 n, および帯状波数 m から $\mathrm{Re}(S_n^m)$ と $\mathrm{Im}(S_n^m)$ の配列中の格納位置を求めるものである.
   - $\mathrm{Im}(S_n^0)$ 成分は存在しないので, $M=0$ の場合は LI には LR と同じ値が返される.

5. 関連項目
   - 関連ルーチン (shtlib)

18.3.4 DclOperateLaplacian

1. 機能

   スペクトルデータに対してラプラシアンを演算する.

2. 書式

```
call DclOperateLaplacian(a, b, [ind], [idx])
```

3. 引数

a	<R(:)>	入力データ.
b	<R(:)>	出力データ.
ind	<I>	ラプラシアンの演算形式.
idx	<I>	作業領域番号.

定義　球面調和関数展開

$$G(\lambda,\varphi)=\sum_{n=0}^{M}\sum_{m=-n}^{n}A_n^m P_n^m(\sin\varphi)e^{im\lambda}. \tag{18.13}$$

に対して、水平 Laplacian

$$\nabla^2\equiv\frac{\partial^2}{\cos^2\varphi\partial\lambda^2}+\frac{\partial}{\cos\varphi\partial\varphi}\left(\cos\varphi\frac{\partial}{\partial\varphi}\right) \tag{18.14}$$

を作用させると, 球面調和関数の性質から,

$$\nabla^2 G(\lambda,\varphi)=\sum_{n=0}^{M}\sum_{m=-n}^{n}-n(n+1)A_n^m P_n^m(\sin\varphi)e^{im\lambda}. \tag{18.15}$$

となる. そこで,

$$B_n^m\equiv -n(n+1)A_n^m \tag{18.16}$$

を導入すると,

$$\nabla^2 G(\lambda,\varphi)=\sum_{n=0}^{M}\sum_{m=-n}^{n}B_n^m P_n^m(\sin\varphi)e^{im\lambda}. \tag{18.17}$$

と表せる. また, 逆に

$$\nabla^2 G(\lambda,\varphi)=\sum_{n=0}^{M}\sum_{m=-n}^{n}A_n^m P_n^m(\sin\varphi)e^{im\lambda}. \tag{18.18}$$

であるとき,

$$B_n^m\equiv -\frac{1}{n(n+1)}A_n^m \tag{18.19}$$

を導入すると,

$$G(\lambda,\varphi)=\sum_{n=0}^{M}\sum_{m=-n}^{n}B_n^m P_n^m(\sin\varphi)e^{im\lambda}. \tag{18.20}$$

と表せる.

本サブルーチンは, IND=1 の場合は A_n^m から $B_n^m \equiv -n(n+1)A_n^m$ を, IND=-1 の場合は A_n^m から $B_n^m \equiv -A_n^m/\{n(n+1)\}$ を計算するものである。

4. 備考
   - ind=-1 の場合, $B_0^0 = 0$ が代入される.
5. 関連項目
   - 関連ルーチン (shtlib)

18.3.5 DclSpectrumToGrid

1. 機能

   スペクトルデータからグリッドデータへ変換する.

2. 書式

```
call DclSpectrumToGrid([s], w, [g], [isw], [idx], [m1], [m2])
```

3. 引数

s	<R(:)>	スペクトルデータ.
w	<R(:)>	ウェーブデータ.
g	<R(:)>	グリッドデータ.
isw	<I>	変換の種類.
idx	<I>	作業領域番号.
m1,m2	<I>	波数空間の最小最大値.

定義 ISW および配列 S, W, G の意味は SHTS2W および SHTW2G に同じである. 本サブルーチンは, SHTS2W, SHTW2G を連続して行うことにより,

ISW=0 の場合, 通常のスペクトル逆変換;

$$G(\lambda,\varphi) = \sum_{n=0}^{M}\sum_{m=-n}^{n} S_n^m P_n^m(\sin\varphi)e^{im\lambda}. \tag{18.21}$$

を行う.

ISW=1 の場合, 緯度微分のスペクトル逆変換;

$$G(\lambda,\varphi) = \frac{\partial}{\partial\varphi}\sum_{n=0}^{M}\sum_{m=-n}^{n} S_n^m P_n^m(\sin\varphi)e^{im\lambda}. \tag{18.22}$$

を行う.

ISW=-1 の場合, 経度微分のスペクトル逆変換;

$$G(\lambda,\varphi) = \frac{\partial}{\partial\lambda}\sum_{n=0}^{M}\sum_{m=-n}^{n} S_n^m P_n^m(\sin\varphi)e^{im\lambda}. \tag{18.23}$$

を行う.

4. 備考

   なし.

5. 関連項目
   - 関連ルーチン (shtlib)

18.3.6 DclGridToSpectrum

1. 機能

   グリッドデータからスペクトルデータへ変換する.

2. 書式

```
call DclGridToSpectrum([g], w, [s], [isw], [idx])
```

3. 引数

g	<R(:)>	グリッドデータ.
w	<R(:)>	ウェーブデータ.
s	<R(:)>	スペクトルデータ.
isw	<I>	変換の種類.
idx	<I>	作業領域番号.

定義 ISW および配列 S, W, G の意味は SHTG2W および SHTW2S に同じである. 本サブルーチンは, SHTG2W, SHTW2S を連続して行うことにより,

ISW=0 の場合, 通常のスペクトル正変換;

$$S_n^m = \frac{1}{4\pi}\int_0^{2\pi}\int_{-\pi/2}^{\pi/2} G(\lambda,\varphi)P_n^m(\sin\varphi)e^{-im\lambda}\cos\varphi d\varphi d\lambda. \tag{18.24}$$

を行う.

ISW=1 の場合, 緯度微分のスペクトル正変換;

$$S_n^m = \frac{1}{4\pi}\int_0^{2\pi}\int_{-\pi/2}^{\pi/2} \frac{\partial}{\cos\varphi\partial\varphi}\left\{\cos\varphi G(\lambda,\varphi)\right\}P_n^m(\sin\varphi)e^{-im\lambda}\cos\varphi d\varphi d\lambda. \tag{18.25}$$

を行う.

ISW=-1 の場合, 経度微分のスペクトル正変換;

$$S_n^m = \frac{1}{4\pi}\int_0^{2\pi}\int_{-\pi/2}^{\pi/2} \frac{\partial}{\cos\varphi\partial\lambda}\left\{G(\lambda,\varphi)\right\}P_n^m(\sin\varphi)e^{-im\lambda}\cos\varphi d\varphi d\lambda. \tag{18.26}$$

を行う.

4. 備考

   なし.

5. 関連項目
   - 関連ルーチン (shtlib)

18.3.7 DclSpectrumToGridForWave

1. 機能

スペクトルデータからグリッドデータへ変換する. (波数指定)

2. 書式

```
call DclSpectrumToGridForWave(m, [s], wr, wi, [g], [isw], [idx])
```

3. 引数

m	<I>	変換する波数.
s	<R(:)>	スペクトルデータ.
wr	<R(:)>	$w^m(\varphi)$ の実数部分.
wi	<R(:)>	$w^m(\varphi)$ の虚数部分.
g	<R(:)>	グリッドデータ.
isw	<I>	変換の種類.
idx	<I>	作業領域番号.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (shtlib)

18.3.8 DclSpectrumToGridForZonal

1. 機能

スペクトルデータからグリッドデータへ変換する. (帯状成分)

2. 書式

```
call DclSpectrumToGridForZonal([s], wz, [g], [isw], [idx])
```

3. 引数

s	<R(:)>	スペクトルデータ.
wz	<R(:)>	$w^0(\varphi)$.
g	<R(:)>	グリッドデータ.
isw	<I>	変換の種類.
idx	<I>	作業領域番号.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (shtlib)

18.3.9 DclSpectrumToGridForLatitude

1. 機能

スペクトルデータからグリッドデータへ変換する. (緯度円指定)

2. 書式

```
call DclSpectrumToGridForLatitude(j, &
[s], wj, [gj], isw, idx, [m1], [m2])
```

3. 引数

j	<I>	変換を行う緯度円.
s	<R(:)>	スペクトルデータ.
wj	<R(:)>	$w^m(\varphi_j)$.
gj	<R(:)>	グリッドデータ.
isw	<I>	変換の種類.
idx	<I>	作業領域番号.
m1,m2	<I>	波数空間の最小最大値.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (shtlib)

18.3.10 DclGetLegendreFunctions

1. 機能

   ルジャンドル陪関数の計算.

2. 書式

```
call DclGetLegendreFunctions(m, fun, [idx])
```

3. 引数

m	<I>	帯状波数.
fun	<R(:)>	ルジャンドル陪関数.
idx	<I>	作業領域番号.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (shtlib)

18.3.11 DclLegendreTransform_F

1. 機能

   ルジャンドル正変換.

2. 書式

```
call DclLegendreTransform_F(m, wm, sm, [isw], [idx])
```

3. 引数

m	<I>	変換を行う帯状波数.
wm	<R(:)>	ウェーブデータ.
sm	<R(:)>	スペクトルデータ.
isw	<I>	変換の種類.
idx	<I>	作業領域番号.

4. 備考

   なし.

5. 関連項目
   - 関連ルーチン (shtlib)

18.3.12 DclLegendreTransform_B

1. 機能

   ルジャンドル逆変換.

2. 書式

```
call DclLegendreTransform_B(m, sm, wm, isw, idx)
```

3. 引数

m	<I>	変換を行う帯状波数.
sm	<R(:)>	スペクトルデータ.
wm	<R(:)>	ウェーブデータ.
isw	<I>	変換の種類.
idx	<I>	作業領域番号.

4. 備考

   なし.

5. 関連項目
   - 関連ルーチン (shtlib)

第 19 章

配列の検索

19.1 ルーチンリスト

DclLocFirst	指定された値が最初に現れる
(INDXNF, INDXIF, INDXRF)	配列の位置を求める.
DclLocLast	指定された値が最後に現れる
(INDXNF, INDXIF, INDXRF)	配列の位置を求める.
DclLocFirstCharEx	指定された文字 (大文字小文字を区別しない)
(INDXMF)	が最初に現れる配列の位置を求める.
DclLocLastCharEx	指定された文字 (大文字小文字を区別しない)
(INDXMF)	が最後に現れる配列の位置を求める.

* 括弧の中は, 対応する f77 インターフェイス名.

19.2 各ルーチンの説明

19.2.1 DclLocFirst

1. 機能

   指定された値が最初に現れる配列の位置を求める.

2. 書式

   ```
   result=DclLocFirst(array, value)
   ```

3. 引数

戻り値	<I>	配列の位置.
array	<R,I,C*(:)>	検索する配列.
value	<R,I,C*>	検索する値.

4. 備考

   - array と value は同じ型でなければならない.

5. 関連項目

   - 関連ルーチン (idxlib)

19.2.2 DclLocLast

1. 機能

   指定された値が最後に現れる配列の位置を求める.

2. 書式

```
result=DclLocLast(array, value)
```

3. 引数

戻り値	<I>	配列の位置.
array	<R,I,C*(:)>	検索する配列.
value	<R,I,C*>	検索する値.

4. 備考

   - array と value は同じ型でなければならない.

5. 関連項目

   - 関連ルーチン (idxlib)

19.2.3 DclLocFirstCharEx

1. 機能

   指定された値 (大文字小文字を区別しない) が最初に現れる配列の位置を求める.

2. 書式

```
result=DclLocFirstCharEx(array, value)
```

3. 引数

戻り値	<I>	配列の位置.
array	<C*(:)>	検索する配列.
value	<C*>	指定する値.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (idxlib)

19.2.4 DclLocLastCharEx

1. 機能

   指定された値 (大文字小文字を区別しない) が最後に現れる配列の位置を求める.

2. 書式

```
result=DclLocLastCharEx(array, value)
```

3. 引数

戻り値	<I>	配列の位置.
array	<C*(:)>	検索する配列.
value	<C*>	指定する値.

4. 備考

なし.

5. 関連項目
   - 関連ルーチン (idxlib)