格子点データ解析ツールGTOOL3
利用の手引

Ver.1.01b

1995 年 5 月 29 日 (暫定)

目 次

第 1 章 概要 **6**
1.1 はじめに . . . 6
1.1.1 GTOOL3 の特徴 . . . 6
1.1.2 GTOOL3 の構成 . . . 6
1.2 ヘッダー . . . 7
1.2.1 ヘッダー一覧 . . . 8
1.2.2 ヘッダーの各記述子の説明 . . . 9
1.3 標準ファイル形式 . . . 10
1.3.1 標準ファイル形式の概要 . . . 10
1.3.2 標準ファイルのフォーマット . . . 11
1.4 格子情報 . . . 11
1.4.1 格子情報の概要 . . . 11
1.4.2 格子情報の種類 . . . 11
1.4.3 格子情報の種類の略称 . . . 12
1.4.4 格子の識別名称 . . . 12
1.4.5 格子情報のフォーマット . . . 12
1.4.6 格子情報のファイル名 . . . 13
1.4.7 格子情報ファイルの生成法 . . . 14
1.5 配列サイズのチェック機能 . . . 14
1.5.1 チェック機能の概要 . . . 14
1.5.2 チェック機能に関する注意 . . . 14
1.6 使用するライブラリ . . . 14
1.7 各モジュールの説明の見方と使い方 . . . 15
1.7.1 モジュールの説明の見方 . . . 15
1.7.2 モジュールの使用法 . . . 16

第 2 章 統括パラメータ管理モジュール群 **17**
2.1 内部パラメータ参照/設定モジュール . . . 17
2.1.1 GTPGET [S] 共通パラメータ（数）を参照 . . . 17
2.1.2 GTPSET [ES] 共通パラメータ（数）を設定 . . . 17
2.1.3 GTCGET [ES] 共通パラメータ（文字）を参照 . . . 18
2.1.4 GTCSET [ES] 共通パラメーター（文字）を設定 . . . 18
2.2 配列サイズ設定モジュール . . . 18
2.2.1 GTSIZE [S] チェック用配列サイズの設定 . . . 18
2.3 作業者名設定モジュール . . . 19
2.3.1 GTSIGN [S] 作業者名の設定 . . . 19

第 3 章 ヘッダー管理モジュール群 20
3.1 ヘッダー参照/設定の基本モジュール 20
3.1.1 GHPGET [S] 記述子（数）の参照 20
3.1.2 GHPSET [ES] 記述子（数）の設定 20
3.1.3 GHCGET [ES] 記述子（文字）の参照 20
3.1.4 GHCSET [ES] 記述子（文字）の設定 21
3.1.5 GHPINQ [ES] 記述子の書式参照 21
3.1.6 GHCGTS [S] 一連の記述子の参照 21
3.1.7 GHCSTS [ES] 一連の記述子の設定 22
3.2 ヘッダーを複写するモジュール 22
3.2.1 GHCOPY [S] ヘッダーの複写 22
3.3 編集記述子を設定するモジュール 22
3.3.1 GHEADD [S] 編集記述子の追加設定 22
3.3.2 GHERST [ES] 編集記述子の再設定 23
3.3.3 GHESET [S] 編集記述子の設定 23
3.3.4 GHQENM [S] 編集記述子の設定数の参照 23
3.4 ヘッダーを一括して参照/設定するモジュール 23
3.4.1 GHPACK [S] データ記述子のパック 23
3.4.2 GHUPAC [ES] 識別記述子のアンパック 25
3.4.3 GHPACA [S] 格子情報ヘッダーパック 26
3.4.4 GHCSQ1 [S] 記述子を書き出しコンソール入力 26
3.4.5 GHCSQ2 [S] 記述子を書き出しコンソール入力 27
3.4.6 GHPTRN [S] ヘッダー形式変換 27
3.4.7 GHRSGP [S] 描画属性記述子のリセット 27
3.4.8 GHPCLR [ES] 記述子のクリアー 27
3.4.9 GHNINQ [ES] 記述子名参照 27

第 4 章 ファイル入出力モジュール群 28
4.1 ファイルの読み書きに関するモジュール 28
4.1.1 GFREAD [S] GTOOL3 標準フォーマットデータ読み込み 28
4.1.2 GFRDSL [S] GTOOL3 標準フォーマットデータ読み込み: 選択つき 28
4.1.3 GFRDHD [ES] GTOOL3 標準フォーマットヘッダー読み込み 29
4.1.4 GFWRIT [S] GTOOL3 標準フォーマットデータ書き込み 29
4.1.5 GFREDZ [S] データ読み込み 29
4.1.6 GFWRTZ [S] データ書き込み 30
4.2 ファイルのオープンに関するモジュール 30
4.2.1 GFROPN [S] ファイル読み込みオープン 30
4.2.2 GFWOPN [S] ファイル書き込みオープン 30
4.2.3 GFCLSE [S] ファイルのクローズ 31
4.2.4 GFOPEN [S] ファイルのオープン 31
4.2.5 GFAOPN [S] ファイル追加書き込みオープン 32
4.2.6 GFOOPN [S] ファイル (追加) 書き込みオープン 32
4.2.7 GFROPQ [S] ファイル名取得＆読み込みオープン 32
4.2.8 GFWOPQ [ES] ファイル名取得＆書き込みオープン 32

4.2.9 GFAOPQ [ES] ファイル名取得&追加書き込みオープン 32
4.2.10 GFOOPQ [ES] ファイル名取得& (追加) 書き込みオープン 33
4.3 そのほかのモジュール 33
4.3.1 GFSADJ [ES] 整合寸法の設定 33
4.3.2 GFQADJ [S] 整合寸法の参照 34
4.3.3 GFWTTS [ES] GTOOL3 標準フォーマット時間設定出力 34
4.3.4 GFRDTS [S] GTOOL3 標準フォーマット入力時間取得 34

第 5 章 データ加工モジュール群 36
5.1 各軸に対する操作のモジュール 36
5.1.1 GPXSEL [S] 第 1 次元の格子選択 36
5.1.2 GPXRDC [S] 第 1 次元の部分切出し 37
5.1.3 GPXAVG [S] 第 1 次元に関する平均 37
5.1.4 GPXEDY [S] 第 1 次元の平均を引く 38
5.1.5 GPXCOM [S] 第 1 次元の結合 38
5.1.6 GPXCMI [S] 第 1 次元の結合の準備 39
5.1.7 GPXEXP [S] 第 1 次元を拡張 39
5.1.8 GPXEXC [ES] 第 1 次元をサイクリックに拡張 40
5.1.9 GPXEXT [S] 第 1 次元の拡大 40
5.2 時系列に関する操作のモジュール 41
5.2.1 GPTAVG [S] 時間平均への足し込み 41
5.2.2 GPTAVO [S] 時間平均の算出 42
5.2.3 GPTAVR [ES] 時間平均のリセット 42
5.2.4 GPTSEQ [S] 時系列の作成 42
5.2.5 GPTSQR [ES] 時系列のリセット 43
5.2.6 GPTSQN [ES] 時系列番号のセット 43
5.3 1 組のデータに対する基本演算操作のモジュール 43
5.3.1 GPFINC [S] データとスカラーの和 43
5.3.2 GPFFCT [S] データとスカラーの積 44
5.3.3 GPFCON [S] データへのスカラー代入 44
5.3.4 GPFSET [S] データにデータを代入 44
5.3.5 GPFFNA [S] データに関数を作用 45
5.4 2 組のデータに対する基本演算操作のモジュール 45
5.4.1 GPFADD [S] データ同士の和 45
5.4.2 GPFSUB [S] データ同士の差 46
5.4.3 GPFMLT [S] データ同士の積 46
5.4.4 GPFDIV [S] データ同士の商 47
5.4.5 GPFFNB [S] 2 つのデータに関数を作用 47
5.5 外部手続き呼びだし汎用モジュール 48
5.5.1 GPCAL1 [S] 外部手続き呼びだし 48
5.5.2 GPCAL2 [S] 2 引数外部手続き呼びだし 49
5.5.3 GPCAL3 [S] 3 引数外部手続き呼びだし 49
5.5.4 GPCAL4 [S] 4 引数外部手続き呼びだし 50

第 6 章 描画モジュール群 52
6.1 基本描画モジュール . . . 52
6.1.1 GGOPEN [S] グラフィック開始 . . . 52
6.1.2 GGCLSE [S] グラフィック終了 . . . 52
6.1.3 GGLAY1 [S] レイアウト, 題名表示 . . . 53
6.1.4 GGAXES [S] 軸のセット X-Y . . . 53
6.1.5 GGAXIS [S] 軸のセット X or Y 軸のみ . . . 54
6.1.6 GGAXRR [ES] X or Y 軸のリセット . . . 54
6.1.7 GGCNTR [S] コンター描画 . . . 54
6.1.8 GGMAPS [S] 地図描画 . . . 55
6.1.9 GGTONE [S] トーン描画 . . . 55
6.1.10 GGCTON [S] トーン段彩 . . . 55
6.1.11 GGSTON [S] トーン段彩設定 . . . 56
6.1.12 GGTONS [S] トーンスケール . . . 56
6.1.13 GGVECT [S] ベクトル描画 . . . 56
6.1.14 GGSCAT [S] 散布図描画 . . . 57
6.1.15 GGCURV [S] 折れ線描画 . . . 57
6.1.16 GGMARK [S] マーク描画 . . . 58
6.1.17 GGLINT [S] 線種タイトル表示 . . . 58
6.1.18 GGRANG [ES] 軸の座標範囲の設定 . . . 59
6.2 描画パラメーター管理モジュール . . . 59
6.2.1 GGPGET [S] 描画パラメータ（数）を参照 . . . 61
6.2.2 GGPSET [ES] 描画パラメータ（数）を設定 . . . 61
6.2.3 GGCGET [ES] 描画パラメータ（文字）を参照 . . . 61
6.2.4 GGCSET [ES] 描画パラメーター（文字）を設定 . . . 61
6.3 描画下請けモジュール . . . 62
6.3.1 GGPDSC [S] ヘッダーの内容 (実数) を文字化して描く . . . 62
6.3.2 GGCDSC [ES] ヘッダーの内容 (文字) を描く . . . 62
6.3.3 GGAXSZ [S] 1 つの軸を描き座標設定 . . . 62
6.3.4 GGAXRS [ES] 軸描画のリセット . . . 63
6.3.5 GGAXLZ [S] 軸ラベルの描画フラグ . . . 63
6.3.6 GGAXIX [S] 軸のセット X or Y 軸のみ：拡張 . . . 63
6.3.7 PORAXZ [S] 外枠囲み (極座標) . . . 63
6.3.8 GGLTRS [ES] 線種タイトル表示リセット . . . 64

第 7 章 ユーティリティーモジュール群 65
7.1 データのサイズの参照等のモジュール . . . 65
7.1.1 GUQSIZ [S] データサイズの取得 . . . 65
7.1.2 GUCSIZ [S] データサイズの比較 . . . 65
7.1.3 GUAXCK [S] 格子名称のチェック . . . 66
7.2 格子情報に関するモジュール . . . 66
7.2.1 GUQAXV [S] 格子情報の取得 . . . 66
7.2.2 GUQAXD [S] 格子のサイズの参照 . . . 67
7.2.3 GUQAXS [S] 格子情報ファイルの読み込み . . . 67

7.2.4 GUEAXN [S] 軸編集記述を作成 . . . 68
7.3 内部設定格子情報に関するモジュール . . . 68
7.3.1 GUSIAX [ES] 内部設定格子の設定 . . . 68
7.3.2 GUAIAX [ES] 内部設定格子位置の付加設定 . . . 69
7.3.3 GUAIAH [ES] 内部設定格子位置の付加設定 (2) . . . 69
7.3.4 GUQIAX [S] 内部設定格子の参照 . . . 70
7.3.5 GUQIAH [ES] 内部設定格子ヘッダーの参照 . . . 70
7.4 配列処理のためのユーティリティーモジュール . . . 71
7.4.1 GUSZCK [S] 配列の大きさのチェック . . . 71
7.4.2 GUSZCZ [S] 大きさのチェック . . . 71
7.4.3 GUSMIS [S] 欠損値を指定 . . . 71
7.5 時刻に関するユーティリティーモジュール . . . 71
7.5.1 GUQTIM [S] 時刻の参照 . . . 71
7.5.2 GUCTIM [S] 時間単位の変換 HUNIT → HUNITY . . . 72
7.5.3 GUQNOW [S] 現在の日付・時刻 yyyymmddhhmmss . . . 72
7.5.4 GUSNOW [S] 現在の時刻を作成時刻に設定 . . . 72
7.5.5 GUTPAC [S] 時刻をパック . . . 73
7.5.6 GUTUPC [ES] 時刻をアンパック . . . 73
7.6 その他のユーティリティーモジュール . . . 73
7.6.1 GUCTXT [S] エスケープシーケンス正規化 . . . 73
7.7 既存パッケージへの追加 . . . 74
7.7.1 GUVINT/GUVDIN/GUVOUT [S] ベクトルデータの重みつき平均 . . . 74

第 8 章 サンプルプログラム 75
8.1 一般的使用法 . . . 75
8.2 コマンド一覧 . . . 75
8.2.1 表示・描画 . . . 75
8.2.2 数学 . . . 76
8.2.3 gtool 形式データ処理 . . . 76
8.2.4 大気物理 . . . 77
8.3 主なコマンドの使用法 . . . 77
8.3.1 gtcont : 2 次元等値線描画, トーン塗分け . . . 77
8.3.2 gtcurv : 折れ線描画 . . . 78
8.3.3 gtshow : データ・ヘッダのキャラクタ表示 . . . 79
8.3.4 gtset : データの線形変換 . . . 79
8.3.5 gtcon : データに一定値をセット . . . 80
8.3.6 gtadd : 二つのデータの和 . . . 81
8.3.7 gtsel : データの切りだし . . . 81
8.3.8 gtext : データの範囲切り出し, 拡張 . . . 82
8.3.9 gtavr : 時間 (系列) 平均 . . . 83
8.3.10 gtseq : 時系列の作成 . . . 83
8.4 コマンドの組み合わせの例 . . . 84

第 9 章 参考文献 85

第1章　概要

1.1　はじめに

格子点データ解析ツール **GTOOL3** は, 格子点によって表されたデータを加工し, 更にそれを図化するための一連の道具である. 小さな道具（以後モジュールと呼ぶ）を組み合わせることによって, さまざまな解析・図化ができるようになることを目指している.

1.1.1　GTOOL3 の特徴

GTOOL3 の特徴として, 次のような点を挙げることができる.

1. ヘッダーによる情報管理

   データの加工・図化に必要なさまざまな情報, 例えばデータの名称, 寸法, 座標軸に関する情報などを, 一つの文字型配列 (ヘッダー配列) に入れて管理する. それによってデータに関する情報の管理と受け渡しが非常に簡単になっている. それらの情報は各モジュールにおいて自動的に参照・変更されるので, あらかじめ正しくセットされていればユーザーがあまり気を使う必要はない.

2. ファイル形式の統一

   データはファイル上には, ヘッダーと共に特定の形式 (**GTOOL3** 標準形式) で書かれる. データ毎に読み取りルーチンを変更する必要はない. データの加工に必要な情報の大部分がファイルに書かれているので, 一つのプログラムでさまざまなデータを扱うことが可能になる.

3. 格子情報ファイルによる格子位置の情報管理

   格子点データにとって, 格子の位置 (座標) は重要な情報である. **GTOOL3** では, 標準的に用いる格子情報 (位置および平均をとる際の重みの) をあらかじめファイル (格子情報ファイル) として作成しておき, 随時それを読み出して利用する. 現在のところ, 3 次元までの, 一般には不等間隔で刻まれた軸によって張られるような直交格子空間のみを扱うことができる.

4. 電脳ライブラリーが基本

   電脳ライブラリー[1] を下位ルーチンとして用いており, 欠損値処理等が可能となっている.

1.1.2　GTOOL3 の構成

GTOOL3 は次のような機能をもつモジュール群に分かれる.

GTTOOL 全体を統括するパラメータを参照/設定する.

GHTOOL ヘッダーを参照/設定する.

GFTOOL 標準形式のファイルを読み書きする.

[1] 文献 [1]

GPTOOL データの加工を行なう.

GGTOOL データの図化を行なう.

GUTOOL 下請け処理モジュールの集まり.

1. **GTTOOL** は, ファイル名称のきまりなど全体にかかわるパラメータを管理する.
2. **GHTOOL** には, ヘッダー配列を操作するためのモジュールがいくつか用意されている.
3. **GFTOOL** には, データをヘッダーと共に **GTOOL3** フォーマットで読み書きするモジュール及びファイルのオープンなどの操作を行なうモジュールが用意されている.
4. **GPTOOL** には, データの切り出し, データの平均, データ同士の四則演算など基本的なモジュールが用意されている. それらをパイプライン的に組み合せることによってさまざまのデータ加工が可能である. SUBFUNC ライブラリを下位ルーチンとして用いている.
5. **GGTOOL** には, モニター的な (必ずしも見た目には整ってはいないが, 必要な情報が多く盛り込まれた) 出力が簡単に出せるようなモジュールが用意されている. ファイルに書かれたヘッダーが自動的に出力されるようになっている. コンター間隔など, 図化に必要なパラメーターもヘッダーの中に記述され, 図化モジュールに渡されるしくみになっている (もちろん内部で適当に決めさせることもできる). UPACK/SGKS を下位ルーチンとして用いている.
6. **GUTOOL** は, 他のモジュールから呼び出される下請けモジュールの集まりである. 中にはユーザーが直接使えるようなものも含まれている.

1.2　ヘッダー

ヘッダーは文字型配列変数 (`CHARACTER(64)*16`) の形をとる. その内容は以下のとおりである. モジュール `GHPACK` (3.4.1) によってセットできる.

1.2.1　ヘッダー一覧

添字	記述子名称	format	説明	例
1	IDFM	I10	フォーマット id	9009
2	DSET	A16	データセット名	AP01
3	ITEM	A16	識別名称 (変数名)	TMP
4	EDIT1	A16	編集略記号 (1)	TM1D
5	EDIT2	A16	編集略記号 (2)	XFK1
⋮	⋮	⋮	⋮	
11	EDIT8	A16	編集略記号 (8)	
12	FNUM	I10	ファイル番号	1
13	DNUM	I10	データ番号	100
14	TITL1	A16	タイトル	Temperature
15	TITL2	A16	〃　続き	
16	UNIT	A16	単位	K
17	ETTL1	A16	編集タイトル (1)	Dayly mean
18	ETTL2	A16	編集タイトル (2)	filter (X)
⋮	⋮	⋮	⋮	
24	ETTL8	A16	編集タイトル (8)	
25	TIME	I10	時刻 (通し)	18769650900
26	DATE	A16	時刻 (yyyymmddhhmmss)	19900813122800
27	UTIM	A16	時刻単位	SEC
28	TDUR	I10	データ代表時間	3600
29	AITM1	A16	軸 1 の格子識別名称	GLON128
30	ASTR1	I10	軸 1 の格子番号始め	1
31	AEND1	I10	軸 1 の格子番号終り	128
32	AITM2	A16	軸 2 の格子識別名称	GGLA64
33	ASTR2	I10	軸 2 の格子番号始め	1
34	AEND2	I10	軸 2 の格子番号終り	64
35	AITM3	A16	軸 3 の格子識別名称	SIGA12
36	ASTR3	I10	軸 3 の格子番号始め	1
37	AEND3	I10	軸 3 の格子番号終り	12
38	DFMT	A16	データフォーマット	(32F12.5) or UR4
39	MISS	E15.7	欠損値の値	-9999.
40	DMIN	E15.7	レンジ (最小)	100.
41	DMAX	E15.7	レンジ (最大)	300.
42	DIVS	E15.7	間隔 (小)	10.
43	DIVL	E15.7	間隔 (大)	50.
44	STYP	I10	スケーリングタイプ	1
45--47	OPTNx	A16	空き	
48--59	MEMOxx	A16	メモ	
60	CDATE	A16	データ作成日付	19900813122800
61	CSIGN	A16	データ作成者	SWAMP
62	MDATE	A16	データ変更日付	19900926225422
63	MSIGN	A16	データ変更者	SWAMP
64	SIZE	I10	配列のサイズ	98304

1.2.2　ヘッダーの各記述子の説明

- IDFM(1) は，今後このヘッダーのフォーマットが変更になった時のために，この形式のデータを別の形式のデータと区別するための情報である． この表はフォーマット id=9009 に基づいている．
- DSET(2) は数値実験なら実験名，観測データならデータの名前 (‘GANAL’ など) である．
- ITEM(3) は例えば変数の略称を入れる．
- EDITx(4∼11) は，編集操作を表す略記号である． はじめは全てブランクであり，データの加工をする度に一つづつ書き加える．
- 以上の DSET(2) から EDIT8(11) までで，ファイル名を構成する．
- FNUM(12) は，ファイルの順序番号となる． 時系列がいくつかのファイルに分かれている場合などに用いる．
- DNUM(13) は，データユニットの順序番号となる．
- TITL1(14)，TITL2(15) は図を描いたりするときのタイトルとなる． 2 つの欄を合わせて 32 文字が使える．
- UNIT(16) は，データの単位を示す．
- ETTLx(17∼24) は，編集操作の内容の簡単な説明である． はじめは全てブランクであり，データの加工をする度に一つづつ書き加える．
- TIME(25) は，適当な時点を原点とした通し時間である．
- DATE(26) は，データの時刻を (yyyymmddhhmmss) という形式で示す．
- UTIM(27) は，TIME(26) および TDUR(28) の単位である．
- TDUR(28) は，データが代表する時間の長さである．
- AITM1(29) は，データの 1 次元めの軸がどの格子情報 (1.4 参照) に対応するかを示す． AITM2，AITM3 等は同様に 2 次元め,3 次元めの指定である．
- ASTR1(30)，AEND1(31) は，データの 1 次元めの軸に格子番号何番から何番までの格子点のデータが入っているかを示す． 通常は ASTR1 は 1，AEND1 はその軸の全格子点数となるが，一部分を切り出したデータの場合はその限りではない． ASTR2，ASTR3 等は同様に 2 次元め,3 次元めの指定である．
- DFMT(38) は，ファイルのデータレコードのフォーマットの指定である． ’(32F12.5)’ などは指定された書式つき，’UR4’ などは書式なし (この場合 4 バイト実数型) を示す．
- MISS(39) は，欠損値扱いするデータの値である．
- MIN(40)，MAX(41)，DIVS(42)，DIVL(43) は図を描くための情報である． これらがセットされている場合，例えばコンターなら MIN から MAX まで DIVS おきにコンターを描き，DIVL おきにラベルをつける． MISS(39) で指定された欠損値が入っている場合には指定されないことになり，内部で適当に決めて描くことになる．

- STYP(44) も図を描くための情報であり，スケーリングタイプと呼ぶ． 折れ線図などのときに，この絶対値が 1 の時は通常の軸で，絶対値が 2 の時は対数軸で描くことを示す． また，正のときは右および上が正の方向，負のときは左および下が正の方向となる．

  すなわち，次のようになる．

	通常の軸	対数軸
右，上が正の方向	1	2
左，下が正の方向	-1	-2

- OPTNx(45〜47，x は 1 から 3 まで) は未使用領域であり，今後の拡張のために使用しないでおく．
- MEMOxx(48〜59，x は 1 から 12 まで) は，メモ欄であり，192 文字の大きさがある． この欄のみ日本語文字を入れてもよい．
- CDATE(60) には，データの作成日付が DATE(26) と同じ形式で入る．
- CSIGN(61) は，データ作成者の署名欄である．
- MDATE(62) には，データの最終変更日付が DATE(26) と同じ形式で入る． データが加工されファイルに書かれるときに，その時刻が入る．
- MSIGN(63) は，データの最終変更者の署名欄である． データが加工されファイルに書かれるときに，モジュール GTSIGN (2.3.1) で設定された作業者名が入る．
- SIZE(64) は，配列の大きさであり，多少特殊な使われかたをする． これが指し示すものは，ファイル上ではデータの大きさに一致するが，計算機上のメモリーに読み込まれてからはデータを入れておく配列の大きさを示し，配列からデータがはみ出すかどうかのチェックに使われる (1.5 参照)．

今後, 各欄のことを ‘記述子’ と呼ぶ. DSET から DNUM までの記述子によってデータに名前が付けられる. これらの記述子を ‘識別記述子’ と呼ぶ. 一方, それ以降の記述子はデータの内容の説明および属性を記述する. これらの記述子を ‘属性記述子’ と呼ぶ. 編集略記号 EDITx および編集タイトル ETTLx をまとめて特別に ‘編集記述子’ と呼ぶ. 編集記述子以外の記述子を ‘データ記述子’ と呼ぶ.

1.3　標準ファイル形式

1.3.1　標準ファイル形式の概要

GTOOL3 標準形式のファイルは ‘データユニット’ の並びで作られる. 1 つのデータユニットは、2 つのレコードからなる. 1 つめは ‘ヘッダーレコード’ であり, そのデータのヘッダーが入る. 2 つめの ‘データレコード’ にデータの本体が入る.

データユニット 1	ヘッダーレコード 1	ファイルの先頭
	データレコード 1	
データユニット 2	ヘッダーレコード 2	
	データレコード 2	
⋮	⋮	
データユニット N	ヘッダーレコード N	
	データレコード N	ファイルの終り

1.3.2　標準ファイルのフォーマット

標準ファイルのフォーマットを示すために, 一つのデータユニットをファイルに書くルーチンを紹介する[2] . ここで, IASTR1*etc.* はヘッダーの ASTR1*etc.* の中身である.

書式なしの場合:

```
 WRITE ( JFILE ) HHEAD
 WRITE ( JFILE ) ( ( ( DATA( I,J,K ), I=IASTR1,IAEND1 ),
&                                    J=IASTR2,IAEND2 ),
&                                    K=IASTR3,IAEND3 )
```

書式ありの場合:

```
 WRITE ( JFILE, '(A)' ) HHEAD
 WRITE ( JFILE, HDFMT ) ( ( ( DATA( I,J,K ), I=IASTR1,IAEND1 ),
&                                           J=IASTR2,IAEND2 ),
&                                           K=IASTR3,IAEND3 )
```

1.4　格子情報

1.4.1　格子情報の概要

座標軸, 格子位置に関する情報は, あらかじめ格子情報として作成しておく. 通常はファイルの形で作成しておくが, メモリー上に作成することもできる. 各モジュールは, ヘッダーから格子の識別名称を認識し, 自動的に格子情報を読み出して利用する. 現在のところ, 3 次元までの, 一般には不等間隔で刻まれた軸によって張られるような直交格子空間のみを扱うことができ, 2 次元以上の空間にランダムに散らばるようなデータは正しく処理できない.

1.4.2　格子情報の種類

格子情報はいくつかの種類に分けられる.

データ内容による分類:

- 格子位置情報
- 格子重み情報

1. 格子位置情報には, 各格子の座標値が入る.
2. 格子重み情報には, その軸に関する平均を取る際の各格子の重みが入る. 和が 1 になっている必要はない. 通常は格子の間隔となるが, 例えば緯度軸の場合には緯線の長さに比例したファクターがかかる.

[2] 一つのファイルに書式つきと書式なしを混在することができないので, データが書式なしの場合はヘッダーも書式なし, データが書式ありの場合はヘッダーも書式ありである必要がある.

今後, 格子位置情報の n 番目の座標に対応する格子 (データ) を, 格子番号 n の格子 (データ) と呼ぶ.

座標軸の性質による分類:

- 周期的な座標軸
- 非周期的な座標軸

周期的な座標軸とは, 例えば経度のように 1 周するともとに戻る座標軸をいう. 周期的な座標軸の場合, 格子番号が (全格子数+1) の格子は格子番号=1 の格子と同じであると仮定される.

データの形態による分類:

- ファイル格子情報
- 内部設定格子情報

ファイル格子情報は, **GTOOL3** 標準形式のファイルに書かれたものである. 通常はファイル格子情報を扱う.
時系列データの時間軸のようなあらかじめ格子位置を特定できない座標軸の場合は, 内部 (メモリー上) に格子情報を蓄えることができる (モジュール GUQIAX/GUSIAX (7.3.4) が管理する) これを内部設定格子情報と呼ぶ.

1.4.3　格子情報の種類の略称

格子情報の種類は, 次のような略称で表す. これは格子情報のヘッダーの欄DSET に入る.

	ファイル格子情報		内部設定格子情報	
	格子位置	格子重み	格子位置	格子重み
周期的	CAXLOC	CAXWGT	CIAXLOC	CIAXWGT
非周期的	AXLOC	AXWGT	IAXLOC	IAXWGT

1.4.4　格子の識別名称

ファイル格子情報の識別名称はどのようなものを付けてもよいが, 互いに区別可能でなければならず, またファイル名の一部となるため (1.4.6 参照) その制約を受ける (HITAC なら先頭が数字でなく, 8 文字以下).
内部設定格子情報の識別名称は, 先頭が文字'@' ではじまるものを付ける.

1.4.5　格子情報のフォーマット

GTOOL3 フォーマットの標準形式で作成される必要がある.
ヘッダーは次のようにセットする (注意すべき項目のみ記す). これはモジュール GHPACA (3.4.3) によってセットできる.

添字	名称	値	例
2	DSET	(格子情報種類略称)	CAXLOC
3	ITEM	(格子の識別名称)	GLON128
12	FNUM	1　(*)	
13	DNUM	1　(*)	
14	TITL1	(軸のタイトル)	longitude
15	TITL2	(　〃　　続き)	
16	UNIT	(軸の単位)	deg
25	TIME	0　(*)	
26	DATE	(作成時刻)	19900813122800
27	UTIM	SEC　(*)	
28	TDUR	1　(*)	
29	AITM1	(ITEM とおなじ)	GLON128
30	ASTR1	1	
31	AEND1	(格子数) または	
		(格子数+1)　[下記]	129
32	AITM2	(空白)	
33	ASTR2	1	
34	AEND2	1	
35	AITM3	(空白)	
36	ASTR3	1	
37	AEND3	1	
40	DMIN	(座標軸を描く際の最小値)	0.
41	DMAX	(座標軸を描く際の最大値)	360.
42	DIVS	(座標軸を描く際の目盛間隔)	10.
43	DIVL	(座標軸を描く際のラベル間隔)	30.
44	STYP	(座標軸のスケーリングタイプ)	1

周期的な座標軸 (識別名称が'C' で始まるもの) の場合は, AEND1 に (格子数+1) を設定する. 非周期的な座標軸 (識別名称が'C' 以外で始まるもの) の場合は, AEND1 に格子数を設定する.
また, このうち (*) を付けたものは必ずしもこれに従う必要はないが慣習としてそれを用いることにするものである. それ以外の記述子は, 上の表に従う必要がある.

1.4.6　格子情報のファイル名

ファイル格子情報の場合のファイル名は,
格子位置情報の場合 :

> 格子位置ファイル接頭子 (モジュール GTCGET/GTCSET (2.1.3) によって管理されるパラメータ FAXLOC の内容) に格子の識別名称をつなげたもの.

格子重み情報の場合 :

> 格子重みファイル接頭子 (モジュール GTCGET/GTCSET (2.1.3) によって管理されるパラメータ FAXWGT の内容) に格子の識別名称をつなげたもの.

となる.

(例えば FAXLOC の内容が'GTAXLOC.', 格子の識別名称が'GLON128' ならば, 格子位置情報のファイル名は 'GTAXLOC.GLON128' である.)

1.4.7　格子情報ファイルの生成法

格子情報ファイルを作成するプログラムの例が付属しているので, 参考にされたい (プログラム gtcrax, ??).

1.5　配列サイズのチェック機能

1.5.1　チェック機能の概要

GTOOL3 では, データは実数型の配列に入れてやりとりする. 実際に何個のデータが入っているかの情報はヘッダーの中に記されており, ファイルにも書かれている. 配列に宣言時の大きさを超えて値を代入しないようにするためには, これらに加えて宣言時の配列のサイズに関する情報を与える必要がある. **GTOOL3** ではこの宣言時の配列のサイズの情報もヘッダーの中に入れて扱う. そのための欄が, 欄 SIZE(64) である. 宣言時の配列のサイズの設定には, モジュール GTSIZE (2.2.1) を用いる. 設定されたサイズよりも多くのデータを書き込もうとしたときには, エラーメッセージを表示して停止する. このチェックはモジュール GUSZCK (7.4.1) によって行なわれる. ただしこのチェック機能は, モジュール GTPGET/GTPSET (2.1.1) の管理するパラメータ SUBCHK が .TRUE. の場合のみ働く. SUBCHK の既定値は.FALSE. である. GTSIZE を呼ぶと, SUBCHK は.TRUE. に設定される.

1.5.2　チェック機能に関する注意

この機能を有効に活用するためには, 次のような注意が必要である.

- ヘッダーを入れる文字型配列と, データを入れる実数型配列は 1 対 1 に対応させておくことが望ましい.
- 宣言部の直後, 最初の実行文として GTSIZE を呼ぶことが望ましい.
- 一つの配列に関して GTSIZE を呼んだときには, 必ず全部の配列について呼ぶこと. GTSIZE によって SIZE が設定されていないときには, 宣言時の配列のサイズが 1 であるとみなされることになるので注意.

ヘッダーの欄 SIZE は, ファイルから読み込まれることはない. また, モジュール GHCOPY (3.2.1) によってもコピーされない.

1.6　使用するライブラリ

GTOOL3 は, 地球流体電脳倶楽部による電脳ライブラリを 下位ルーチンとして用いている. この利用の手引は電脳ライブラリに関して多少の知識があることを前提として書かれている. 電脳ライブラリに関しては文献 [1] を参照のこと.

1.7　各モジュールの説明の見方と使い方

1.7.1　モジュールの説明の見方

第 2 章から第 7 章までは, 各モジュールの説明にあてられる. そこでは, 次のような説明がある.

2.1.1　GTPGET [S] 共通パラメータ（数）を参照

```
      SUBROUTINE GTPGET
     I         ( HP,
     O           IPARA )
*
      CHARACTER HP    *(*)                ! パラメーターの名前
*     (INTEGER)  IPARA                    ! パラメーターの内容：数字
```

パラメーター HP の内容を IPARA に入れる．　IPARA の実引数としては，適当な型の変数を用いること．

タイトルは, モジュール名とその簡単な説明である. [S] などの記号は, モジュールの種類であり,

[P]	プログラム
[S]	サブルーチン
[ES]	サブルーチンへのエントリー
[F]	関数
[EF]	関数へのエントリー
[I]	INCLUDE で読み込まれる部分

となっている.

次に, モジュールの先頭部分 (モジュール名, 仮引数のリスト, 宣言部のうち仮引数に関係する部分) を実際のプログラムからコピーしてある. 宣言部には簡単なコメントが書いてあり, それによって引数の内容と型を示す.
仮引数リストの継続行の記号は次のような意味を持っている.

記号	意味	入力	出力	機能
O	output	×	○	値を生成
M	modify	○	○	入力値を加工して出力
I	input	○	−	入力値
D	dimension	○	−	整合配列の大きさを決める
W	work	×	×	作業領域

入力
- ○　中で値が参照される可能性がある
- ×　中に何が入っているかは問わない

出力
- ○　中で値が変更される可能性がある
- −　値は変更されない
- ×　何がでてくるかは保証しない

最後にモジュールの内容についての多少くわしい説明がある.

1.7.2　モジュールの使用法

説明に書かれた情報をもとに, 適当な引数を用いてモジュールを CALL すればよい.

ただし, 次の記述には多少注意が必要である.

```
*    (INTEGER)  IPARA                    ! パラメーターの内容：数字
```

この仮引数は, モジュールの中では整数型として宣言されているが, 実引数としては場合によって整型数/実数型/論理型のどれでも取りうる. これがどの型を取るかは, 別の引数 (ここでは HP) によって指定されることになる. (モジュールの中では EQUIVALENCE 文によって整型数/実数型/論理型の変数が結合されて表現されている.) このような引数を持つモジュールには, モジュール GTPGET/GTPSET (2.1.1) モジュール GHPGET/GHPSET (3.1.1) モジュール GGPGET/GGPSET (6.2.1) がある.
これらのモジュールを呼ぶ際には, 引数の型に注意しなければならない. 例えば, GTPGET の場合, HP が 'WFILE' のときは IPARA として整数型の変数/定数を用いるべきだが, HP が 'SUBCHK' のときは IPARA として論理型の変数/定数を用いなければならない. また, コンパイルオプションとして ARGCHK オプションを用いてデバッグを行なうときは, これらのモジュールについてだけは, NOARGCHK オプションを用いてコンパイルしておかないと (本当は害のない) エラーが出てしまう.

第2章　統括パラメータ管理モジュール群

全体にかかわるパラメータ等を管理するモジュール群である.

2.1　内部パラメータ参照/設定モジュール

次のような内部パラメータを管理している.

パラメーター名	種類	既定値	説明
MISS	R	-999.	デフォルトの欠損値
SUBCHK	L	.FALSE.	配列の大きさのチェックを行なうか否か
WFILE	I	91	作業ファイル装置番号．　格子情報の読み込みに使用
STYTSQ	I	1	時間軸のスケーリングタイプ．　時系列作成時に使用
FAXLOC	C	'GTAXLOC.'	格子位置ファイル接頭子
FADLOC	C	'GTAXWGT.'	格子重みファイル接頭子
UTIM	C	'DAY '	時間表示の単位．　図化や時系列の作成のときに使用
MYSIGN	C	'GTOOL3'	作業者名

数字/論理パラメータに関してはモジュール GTPGET/GTPSET (2.1.1) で, 文字パラメータに関してはモジュール GTCGET/GTCSET (2.1.3) で, それぞれ参照/設定できる.

2.1.1　GTPGET [S] 共通パラメータ（数）を参照

```
      SUBROUTINE GTPGET
     I         ( HP,
     O           IPARA )
*
      CHARACTER HP    *(*)                ! パラメーターの名前
*     (INTEGER)  IPARA                    ! パラメーターの内容：数字
```

パラメーター HP の内容を IPARA に入れる．　IPARA の実引数としては，適当な型の変数を用いること (1.7.2 参照).

2.1.2　GTPSET [ES] 共通パラメータ（数）を設定

```
      ENTRY GTPSET
     I         ( HP, IPARA )
*
      CHARACTER HP    *(*)                ! パラメーターの名前
*     (INTEGER)  IPARA                    ! パラメーターの内容：数字
```

IPARA をパラメーター HP に入れる． IPARA の実引数としては，適当な型の変数/定数を用いること (1.7.2 参照)．

2.1.3　GTCGET [ES] 共通パラメータ（文字）を参照

```
      ENTRY GTCGET
     I         ( HP,
     O           HPARA )
*
      CHARACTER HP   *(*)              ! パラメーターの名前
      CHARACTER HPARA*(*)              ! パラメーターの内容：文字
*
```

パラメーター HP の内容を HPARA に入れる．

2.1.4　GTCSET [ES] 共通パラメーター（文字）を設定

```
      ENTRY GTCSET
     I         ( HP, HPARA )
*
      CHARACTER HP   *(*)              ! パラメーターの名前
      CHARACTER HPARA*(*)              ! パラメーターの内容：文字
*
```

HPARA をパラメーター HP に入れる．

2.2　配列サイズ設定モジュール

モジュール GTPGET/GTPSET (2.1.1) の管理するパラメータ SUBCHK (既定値は .FALSE.) が .TRUE. の時は, 配列の大きさのチェックが有効になる. チェックが正常に働くためには, 配列の大きさの情報をヘッダーの中にあらかじめ設定しておく必要がある. くわしくは 1.5 節を参照のこと.

2.2.1　GTSIZE [S] チェック用配列サイズの設定

```
      SUBROUTINE GTSIZE
     M         ( HHEAD ,
     I           ISIZE  )
*
      CHARACTER  HHEAD  ( * )*(*)      ! ヘッダー
      INTEGER    ISIZE                 ! 配列サイズ
```

配列の大きさを越えて値を代入するのをチェックする際に使用するために，配列の大きさを HHEAD の欄 SIZE に設定する． さらにモジュール GTPGET/GTPSET (2.1.1) の管理するパラメータ SUBCHK (既定値は .FALSE.) を .TRUE. に設定し，チェックを有効にする．

2.3　作業者名設定モジュール

データが加工されファイルに書かれるときには，ヘッダーの記述子 MDATE(62) にその時刻が，記述子 MSIGN(63) に作業者名が入る．この作業者名の設定を行なうのがこのモジュールである．作業者名が設定されていない場合にファイルに書き込もうとすると警告が出る. データの管理上, ぜひとも作業者名を設定して **GTOOL3** を使うようにすべきである.

2.3.1　GTSIGN [S] 作業者名の設定

```
      SUBROUTINE GTSIGN
     I         ( HSIGN  )
*
      CHARACTER  HSIGN       *(*)        ! 名前
```

モジュール GTPGET/GTPSET (2.1.1) の管理するパラメータ MYSIGN (既定値は 'GTOOL3') に指定した文字列を入れる.

第3章　ヘッダー管理モジュール群

ヘッダーを参照/設定するモジュールが用意されている.

3.1　ヘッダー参照/設定の基本モジュール

3.1.1　GHPGET [S] 記述子（数）の参照

```
      SUBROUTINE GHPGET
     I         ( HHEAD , HP    ,
     I           IX              )
*
      INCLUDE    (GZSIZE)                ! NCC, NDC
      CHARACTER  HHEAD (NDC)*(NCC)       ! ヘッダー
      CHARACTER  HP         *(*)         ! ヘッダーの記述子指定
*     (INTEGER)  IX                      ! 記述子の内容：数
```

HHEAD の記述子 HP に設定されている情報を数データとして参照する． 数データ記述子でないもの（‘A16’指定の記述子）に対しては「-1」を返す． IX の実引数としては，適当な型の変数を用いること（1.7.2 参照）.

3.1.2　GHPSET [ES] 記述子（数）の設定

```
      ENTRY GHPSET
     M         ( HHEAD ,
     I           HP    , IX     )
*
      INCLUDE    (GZSIZE)                ! NCC, NDC
      CHARACTER  HHEAD (NDC)*(NCC)       ! ヘッダー
      CHARACTER  HP         *(*)         ! ヘッダーの記述子指定
*     (INTEGER)  IX                      ! 記述子の内容：数
```

HHEAD の記述子（記述子)HP に情報を数データとして設定する． 数データ記述子でないもの（‘A16’指定の記述子）に対して設定しようとしても無視される． IX の実引数としては，適当な型の変数/定数を用いること（1.7.2 参照）.

3.1.3　GHCGET [ES] 記述子（文字）の参照

```
      ENTRY      GHCGET
```

```
      I         ( HHEAD , HP    ,
      I           HX              )
*
       INCLUDE    (GZSIZE)                ! NCC, NDC
       CHARACTER  HHEAD (NDC)*(NCC)       ! ヘッダー
       CHARACTER  HP         *(*)         ! ヘッダーの記述子指定
       CHARACTER  HX         *(*)         ! 記述子の内容：文字
```

HHEAD の記述子 HP に設定されている情報を文字データとして取得する.

3.1.4　GHCSET [ES] 記述子（文字）の設定

```
       ENTRY GHCSET
      M         ( HHEAD ,
      I           HP    , HX     )
*
       INCLUDE    (GZSIZE)                ! NCC, NDC
       CHARACTER  HHEAD (NDC)*(NCC)       ! ヘッダー
       CHARACTER  HP         *(*)         ! ヘッダーの記述子指定
       CHARACTER  HX         *(*)         ! 記述子の内容：文字
```

HHEAD の記述子 HP に情報を文字データとして設定する． 16 文字を超える分は切り捨てられる.

3.1.5　GHPINQ [ES] 記述子の書式参照

```
       ENTRY GHPINQ
      I         ( HP    ,
      O           HFM    )
*
       CHARACTER  HP         *(*)         ! ヘッダーの記述子指定
       CHARACTER  HFM        *(*)         ! 記述子の書式
```

記述子 HP の書式を参照する． 記述子 HP が存在しないときは空白 (' ') が返される.

3.1.6　GHCGTS [S] 一連の記述子の参照

```
       SUBROUTINE GHCGTS
      I         ( HHEAD, HP    ,
      O           HX            )
*
       CHARACTER  HHEAD ( * )*(*)         ! ヘッダー
       CHARACTER  HP         *(*)         ! ヘッダーの記述子指定
       CHARACTER  HX         *(*)         ! 記述子の内容：文字
```

HP の内容に順次 1,2,3,...を付けた名前の記述子（HP='TITL' なら,TITL1,TITL2）に設定されている情報を結合して文字データとして参照する．

3.1.7　GHCSTS [ES] 一連の記述子の設定

```
      ENTRY      GHCSTS
     I         ( HHEAD, HP    ,
     I           HX            )
*
      CHARACTER  HHEAD ( * )*(*)         ! ヘッダー
      CHARACTER  HP          *(*)         ! ヘッダーの記述子指定
      CHARACTER  HX          *(*)         ! 記述子の内容：文字
```

HP の内容に順次 1,2,3,...を付けた名前の記述子（HP='TITL' なら,TITL1,TITL2）に，文字データを分割して設定する． HX の長さが記述子を結合した全体の長さ（上の例では TITL1 と TITL2 を足した長さ）に満たない場合には残りには空白が入る．

3.2　ヘッダーを複写するモジュール

3.2.1　GHCOPY [S] ヘッダーの複写

```
      SUBROUTINE GHCOPY
     I         ( HHEAD ,
     O           HHEADO )
*
      CHARACTER  HHEAD ( * )*(*)         ! ヘッダー
      CHARACTER  HHEADO( * )*(*)         ! ヘッダー（出力）
```

ヘッダーの複製を作成する． ただし，項目 SIZE(64) は複写されない．

3.3　編集記述子を設定するモジュール

3.3.1　GHEADD [S] 編集記述子の追加設定

```
      SUBROUTINE GHEADD
     M         ( HHEAD ,
     I           HEDIT , HETTL )
*
      CHARACTER  HHEAD ( * )*(*)         ! ヘッダー
      CHARACTER  HEDIT *(*)              ! 編集略記号
      CHARACTER  HETTL *(*)              ! 編集タイトル
```

現在設定されている編集記述子の後に，新しい編集記述子（HEDIT と HETTL）を付け加える．編集記述領域がいっぱいの場合は，警告を出し，設定されない．

3.3.2　GHERST [ES] 編集記述子の再設定

```
      ENTRY      GHERST
     M         ( HHEAD ,
     I           HEDIT , HETTL )
*
      CHARACTER  HHEAD ( * )*(*)         ! ヘッダー
      CHARACTER  HEDIT *(*)              ! 編集略記号
      CHARACTER  HETTL *(*)              ! 編集タイトル
      INCLUDE    (GZSIZE)                ! NCC, NDC, NED
```

現在設定されている一番最後の編集記述子を，新しい編集記述子（HEDIT と HETTL）に入れ換える．

3.3.3　GHESET [S] 編集記述子の設定

```
      SUBROUTINE GHESET
     M         ( HHEAD ,
     I           HEDIT , HETTL, IENUM  )
*
      CHARACTER  HHEAD ( * )*(*)         ! ヘッダー
      CHARACTER  HEDIT *(*)              ! 編集略記号
      CHARACTER  HETTL *(*)              ! 編集タイトル
```

現在設定されているいないにかかわらず，IENUM 番の編集記述子（HEDIT と HETTL）を設定する．

3.3.4　GHQENM [S] 編集記述子の設定数の参照

```
      SUBROUTINE GHQENM
     I         ( HHEAD ,
     O           IENUM )
*
      CHARACTER  HHEAD ( * )*(*)         ! ヘッダー
      INTEGER    IENUM                   ! 設定済の編集記述子の数
```

現在設定されている編集記述子の数を参照する．

3.4　ヘッダーを一括して参照/設定するモジュール

3.4.1　GHPACK [S] データ記述子のパック

```
      SUBROUTINE GHPACK
     O         ( HHEAD ,
     I           HDSET , HITEM , IFNUM , IDNUM ,
```

```
      I           HTITL , HUNIT ,
      I           ITIME , HDATE , HUTIM , ITDUR ,
      I           HAITM1, IASTR1, IAEND1,
      I           HAITM2, IASTR2, IAEND2,
      I           HAITM3, IASTR3, IAEND3,
      I           HDFMT , VMISS ,
      I           DMIN  , DMAX  , DIVS  , DIVL  , ISTYP ,
      I           HCDATE, HCSIGN                        )
*
       CHARACTER  HHEAD ( * )*(*)
       CHARACTER  HDSET  *(*)            ! データセット名      DSET
       CHARACTER  HITEM  *(*)            ! 識別名称            ITEM
       INTEGER    IFNUM                  ! ファイル番号        FNUM
       INTEGER    IDNUM                  ! データ番号          DNUM
       CHARACTER  HTITL  *(*)            ! タイトル            TITL
       CHARACTER  HUNIT  *(*)            ! 単位                UNIT
       INTEGER    ITIME                  ! 時刻 (通し)         TIME
       CHARACTER  HDATE  *(*)            ! 時刻                DATE
       CHARACTER  HUTIM  *(*)            ! 時刻単位            UTIM
       INTEGER    ITDUR                  ! 代表する時間        TDUR
       CHARACTER  HAITM1 *(*)            ! 軸 1 の格子名称     AITM1
       INTEGER    IASTR1                 ! 軸 1 の格子始め     ASTR1
       INTEGER    IAEND1                 ! 軸 1 の格子終り     AEND1
       CHARACTER  HAITM2 *(*)            ! 軸 2 の格子名称     AITM1
       INTEGER    IASTR2                 ! 軸 2 の格子始め     ASTR1
       INTEGER    IAEND2                 ! 軸 2 の格子終り     AEND1
       CHARACTER  HAITM3 *(*)            ! 軸 3 の格子名称     AITM1
       INTEGER    IASTR3                 ! 軸 3 の格子始め     ASTR1
       INTEGER    IAEND3                 ! 軸 3 の格子終り     AEND1
       CHARACTER  HDFMT  *(*)            ! データフォーマット  DFMT
       REAL       VMISS                  ! 欠損値の値          MISS
       REAL       DMIN                   ! レンジ (最小)       DMIN
       REAL       DMAX                   ! レンジ (最大)       DMAX
       REAL       DIVS                   ! 間隔 (小)           DIVL
       REAL       DIVL                   ! 間隔 (大)           DIVS
       INTEGER    ISTYP                  ! スケーリングタイ プ STYP
       CHARACTER  HCDATE *(*)            ! データ作成日付      CDATE
       CHARACTER  HCSIGN *(*)            ! データ作成者        CSIGN
```

HHAED に，編集記述子以外の各データ記述子の情報を設定する． HCDATE に空白 (' ') を指定した場合には，CDATE に現在の時刻が入る． MDATE および MSIGN には，現在の時刻およびモジュール GTPGET/GTPSET (2.1.1) の管理するパラメータ MYSIGN (既定値は 'GTOOL3') が入る．

3.4.2　GHUPAC [ES] 識別記述子のアンパック

```
      ENTRY     GHUPAC
     I        ( HHEAD ,
     O          HDSET , HITEM , IFNUM , IDNUM ,
     O          HTITL , HUNIT ,
     O          ITIME , HDATE , HUTIM , ITDUR ,
     O          HAITM1, IASTR1, IAEND1,
     O          HAITM2, IASTR2, IAEND2,
     O          HAITM3, IASTR3, IAEND3,
     O          HDFMT , VMISS ,
     O          DMIN  , DMAX  , DIVS  , DIVL  , ISTYP ,
     O          HCDATE, HCSIGN, HMDATE, HMSIGN        )
*
      CHARACTER  HHEAD ( * )*(*)
      CHARACTER  HDSET  *(*)            ! データセット名      DSET
      CHARACTER  HITEM  *(*)            ! 識別名称            ITEM
      INTEGER    IFNUM                  ! ファイル番号        FNUM
      INTEGER    IDNUM                  ! データ番号          DNUM
      CHARACTER  HTITL  *(*)            ! タイトル            TITL
      CHARACTER  HUNIT  *(*)            ! 単位                UNIT
      INTEGER    ITIME                  ! 時刻（通し）        TIME
      CHARACTER  HDATE  *(*)            ! 時刻                DATE
      CHARACTER  HUTIM  *(*)            ! 時刻単位            UTIM
      INTEGER    ITDUR                  ! 代表する時間        TDUR
      CHARACTER  HAITM1 *(*)            ! 軸 1 の格子名称     AITM1
      INTEGER    IASTR1                 ! 軸 1 の格子始め     ASTR1
      INTEGER    IAEND1                 ! 軸 1 の格子終り     AEND1
      CHARACTER  HAITM2 *(*)            ! 軸 2 の格子名称     AITM1
      INTEGER    IASTR2                 ! 軸 2 の格子始め     ASTR1
      INTEGER    IAEND2                 ! 軸 2 の格子終り     AEND1
      CHARACTER  HAITM3 *(*)            ! 軸 3 の格子名称     AITM1
      INTEGER    IASTR3                 ! 軸 3 の格子始め     ASTR1
      INTEGER    IAEND3                 ! 軸 3 の格子終り     AEND1
      CHARACTER  HDFMT  *(*)            ! データフォーマット  DFMT
      REAL       VMISS                  ! 欠損値の値          MISS
      REAL       DMIN                   ! レンジ（最小）      DMIN
      REAL       DMAX                   ! レンジ（最大）      DMAX
      REAL       DIVS                   ! 間隔（小）          DIVL
      REAL       DIVL                   ! 間隔（大）          DIVS
      INTEGER    ISTYP                  ! スケーリングタイプ  STYP
      CHARACTER  HCDATE *(*)            ! データ作成日付      CDATE
      CHARACTER  HCSIGN *(*)            ! データ作成者        CSIGN
      CHARACTER  HMDATE *(*)            ! データ変更日付      MDATE
```

```
      CHARACTER  HMSIGN *(*)             ! データ変更者        MSIGN
```

HHAED から編集記述子以外の各データ記述子の情報を取り出す.

3.4.3　GHPACA [S] 格子情報ヘッダーパック

```
       SUBROUTINE GHPACA
      O         ( HHEADA,
      I           HKIND , HITEM ,
      I           HTITL , HUNIT ,
      I           IXDIM ,
      I           HDFMT , VMISS ,
      I           DMIN  , DMAX  , DIVS  , DIVL  , ISTYP  )
*
      CHARACTER  HHEADA ( * )*(*)        ! ヘッダー
      CHARACTER  HKIND       *(*)        ! 格子情報の種類      DSET
      CHARACTER  HITEM       *(*)        ! 格子識別名称        ITEM, AITM1
      CHARACTER  HTITL       *(*)        ! 軸タイトル          TITL
      CHARACTER  HUNIT       *(*)        ! 単位                UNIT
      INTEGER    IXDIM                   ! 格子数              AEND1
      CHARACTER  HDFMT       *(*)        ! データフォーマット  DFMT
      REAL       VMISS                   ! 欠損値の値          MISS
      REAL       DMIN                    ! レンジ (最小)       DMIN
      REAL       DMAX                    ! レンジ (最大)       DMAX
      REAL       DIVS                    ! 間隔 (小)           DIVS
      REAL       DIVL                    ! 間隔 (大)           DIVL
      INTEGER    ISTYP                   ! スケーリングタイプ  STYP
```

HHEADA に，格子情報ファイル用のヘッダーを作成する.

HKIND には，格子情報の種類の表 (1.4.3) にあるもののうち一つを与えること. IXDIM には，周期的な軸・非周期的な軸にかかわらず全格子数を与えること. 1.4.5 参照.

3.4.4　GHCSQ1 [S] 記述子を書き出しコンソール入力

```
       SUBROUTINE GHCSQ1
      I         ( HH1   , HP    , IC    ,
      M           HHO                   )
*
      CHARACTER  HH1  ( * )*(*)
      CHARACTER  HHO  ( * )*(*)
      CHARACTER  HP        *(*)
      INTEGER    IC                     !"記憶番号
```

3.4.5　GHCSQ2 [S] 記述子を書き出しコンソール入力

```
      SUBROUTINE GHCSQ2
     I         ( HH1   , HH2   , HP    , IC    ,
     M           HHO                             )
*
      CHARACTER  HH1  ( * )*(*)
      CHARACTER  HH2  ( * )*(*)
      CHARACTER  HHO  ( * )*(*)
      CHARACTER  HP        *(*)
      INTEGER    IC                      !" 記憶番号
```

3.4.6　GHPTRN [S] ヘッダー形式変換

```
      SUBROUTINE GHPTRN
     M         ( HHEAD  )
*
      CHARACTER  HHEAD (NDC)*(NCC)       !" ヘッダー
```

3.4.7　GHRSGP [S] 描画属性記述子のリセット

```
      SUBROUTINE GHRSGP
     I         ( HHEAD )
```

3.4.8　GHPCLR [ES] 記述子のクリアー

```
      ENTRY GHPCLR
     O         ( HHEAD  )
*
      CHARACTER  HHEAD (NDC)*(NCC)       !" ヘッダー
```

3.4.9　GHNINQ [ES] 記述子名参照

```
      ENTRY GHNINQ
     I         ( IZ    ,
     O           HP      )
*
      CHARACTER  HP         *(*)         !" ヘッダーの記述子指定
```

第4章　ファイル入出力モジュール群

ファイルの入出力に関するモジュールが用意されている. ファイルは基本的には装置番号で指定する.

4.1　ファイルの読み書きに関するモジュール

4.1.1　GFREAD [S] GTOOL3 標準フォーマットデータ読み込み

```
      SUBROUTINE GFREAD
     O         ( HHEAD , GDATA , IEOD  ,
     I           IFILE , MODE          )
*
      CHARACTER  HHEAD ( * )*(*)         ! ヘッダー
      REAL       GDATA ( * )             ! データ
      INTEGER    IEOD                    ! データ存在フラグ
      INTEGER    IFILE                   ! 装置番号
      INTEGER    MODE                    ! モード（1 なら表示 ON）
```

装置番号 IFILE のファイルの，現在指しているデータユニットを読み込む． データが存在しないときには，IEOD として 0 でない値を返す.

MODE=1 と指定すると，読んだユニットのヘッダーの ITEM と TIME を表示する.

4.1.2　GFRDSL [S] GTOOL3 標準フォーマットデータ読み込み: 選択つき

```
      SUBROUTINE GFRDSL
     O         ( HHEAD , GDATA , IEOD  ,
     I           HPSEL , HSEL  , ISEL  ,
     I           IFILE , MODE          )
*
      CHARACTER  HHEAD ( * )*(*)         ! ヘッダー
      REAL       GDATA ( * )             ! データ
      INTEGER    IEOD                    ! データ存在フラグ
      CHARACTER  HPSEL *(*)              ! 選択する欄
      CHARACTER  HSEL  *(*)              ! 選択するデータ（文字）
      INTEGER    ISEL                    ! 選択するデータ（数）
      INTEGER    IFILE                   ! 装置番号
      INTEGER    MODE                    ! モード（1 なら表示 ON）
```

装置番号 IFILE のファイルの，現在指しているデータユニット以降で，ヘッダーの欄 HPSEL の内容が HSEL(文字) または ISEL(数) に等しいものを探して，最初のものを読み込む． HPSEL に'###' を指定すると，無条件に読み込む (モジュール GFREAD (4.1.1) と同じ)． データが存在しないときには，IEOD として 0 でない値を返す．

MODE=1 とすると，読んだユニットのヘッダーの ITEM と TIME を表示する．

4.1.3　GFRDHD [ES] GTOOL3 標準フォーマットヘッダー読み込み

```
      ENTRY     GFRDHD
     O         ( HHEAD , IEOD  ,
     I           IFILE , MODE   )
*
      CHARACTER  HHEAD ( * )*(*)        ! ヘッダー
      INTEGER    IEOD                   ! データ存在フラグ
      REAL       GDATA ( * )            ! データ
      INTEGER    IFILE                  ! 装置番号
      INTEGER    MODE                   ! モード（1 なら表示 ON）
```

装置番号 IFILE のファイルの，現在指しているデータユニットのヘッダーレコードを読み，データレコードを読みとばす． データが存在しないときには，IEOD として 0 でない値を返す．

MODE=1 と指定すると，読んだユニットのヘッダーの ITEM と TIME を表示する．

4.1.4　GFWRIT [S] GTOOL3 標準フォーマットデータ書き込み

```
      SUBROUTINE GFWRIT
     I         ( HHEAD , GDATA ,
     I           JFILE , MODE  , NOEND   )
*
      CHARACTER  HHEAD ( * )*(*)        ! ヘッダー
      REAL       GDATA ( * )            ! データ
      INTEGER    JFILE                  ! 装置番号
      INTEGER    MODE                   ! モード（1 なら表示 ON）
      INTEGER    NOEND                  ! ENDFILE の間隔
```

装置番号 IFILE のファイルに，データユニットを追加して書き込む．

NOEND≠0 の時は，同じ装置番号のファイルに NOEND 個のデータユニットを書き込む毎に ENDFILE[1] 処理を行なう． NOEND=0 の時は ENDFILE 処理を行なわない．

MODE=1 とすると，書き込んだユニットのヘッダーの ITEM と TIME を表示する．

4.1.5　GFREDZ [S] データ読み込み

```
      SUBROUTINE GFREDZ
```

[1] ENDFILE 処理を行なうと, HITAC では操作対象となる DD 名 FTxxFyyy の yyy が一つ増える

```
      O         ( GDATA , IEOD  ,
      I           IFILE , IJKDIM )
```

4.1.6　GFWRTZ [S] データ書き込み

```
       SUBROUTINE GFWRTZ
      I         ( GDATA ,
      I           JFILE , IJKDIM )
*
       REAL       GDATA ( IJKDIM )              !" データ
       INTEGER    JFILE                         !" ファイル番号
```

4.2　ファイルのオープンに関するモジュール

4.2.1　GFROPN [S] ファイル読み込みオープン

```
       SUBROUTINE GFROPN
      M         ( IFILE ,
      I           HFILE  )
*
       INTEGER    IFILE                    ! 装置番号
       CHARACTER  HFILE*(*)                ! ファイル名
```

読み込み専用にファイルをオープンし，装置番号 IFILE 番に割り当てる． IFILE 番がすでに接続されている場合には，IFILE 番以降で空いている装置番号を捜して割り当て，割り当てた装置番号を IFILE に返す．

ファイル名として数字（例えば HFILE='20'）を指定した場合は，その装置番号がすでにファイルに接続されていると見なして，何もせずにファイル名を整数に変換して IFILE に返す． ファイル名として，'CON' を指定した場合は，5 を IFILE として返す．

ファイルの形式は順データセット，書式なしである． ファイルが存在しなかった場合はエラーを表示し停止する．

4.2.2　GFWOPN [S] ファイル書き込みオープン

```
       SUBROUTINE GFWOPN
      M         ( IFILE ,
      I           HFILE  )
*
       INTEGER    IFILE                    ! 装置番号
       CHARACTER  HFILE*(*)                ! ファイル名
*
```

書き込み専用にファイルをオープンし，装置番号 IFILE 番に割り当てる． IFILE 番がすでに接続されている場合には，IFILE 番以降で空いている装置番号を捜して割り当て，割り当てた装置番号を IFILE に返す．

ファイル名として数字（例えば HFILE='20'）を指定した場合は，その装置番号がすでにファイルに接続されていると見なして，何もせずにファイル名を整数に変換して IFILE に返す． ファイル名として，'CON' を指定した場合は，6 を IFILE として返す．

ファイルの形式は順データセット，書式なしである． オープンエラーが発生した場合はエラーを表示し停止する．

4.2.3　GFCLSE [S] ファイルのクローズ

```
      SUBROUTINE GFCLSE
     M         ( IFILE  )
*
      INTEGER    IFILE                    ! 装置番号
```

装置番号 IFILE のファイルをクローズする． IFILE として，1〜99 以外のものを指定したときには何もしない．

4.2.4　GFOPEN [S] ファイルのオープン

```
      SUBROUTINE GFOPEN
     O         ( IFILE , IERR  ,
     I           HFILE , IFILED, HACT  , HFORM   )
*
      INTEGER    IFILE                      ! 装置番号
      INTEGER    IERR                       ! エラー≠0
      CHARACTER  HFILE *(*)                 ! ファイル名
      INTEGER    IFILED                     ! デフォルト装置番号
      CHARACTER  HACT  *(*)                 ! 読み／書き
      CHARACTER  HFORM *(*)                 ! フォーマット
```

ファイルをオープンし，装置番号 IFILED 番に割り当てる． IFILED 番がすでに接続されている場合には，IFILED 番以降で空いている装置番号を捜して割り当てる． 割り当てた装置番号を IFILE に返す．

ファイル名として数字（例えば HFILE='20'）を指定した場合は，その装置番号がすでにファイルに接続されていると見なして，何もせずにファイル名を整数に変換して IFILE に返す． ファイル名として，'CON' を指定した場合は，読み込み用の場合は 5 を，書き込み用の場合は 6 を IFILE として返す．

ファイルの形式は順データセットとする． HACTが'READ'の場合は読み込み専用，HACTが'WRITE'の場合は書き込み専用となる． HFORMが'FORMATTED'の場合は書式つき，HFORMが'UNFORMATTED'の場合は書式なしとしてオープンする．

オープンエラーが発生した場合は，エラーコード（0 以外）を IERR に返す． 読み込み用にオープンする場合でファイルが存在しないときはエラーコード 1 を返す． エラーが起きた場合，IFILE には 0 が入る．

4.2.5 GFAOPN [S] ファイル追加書き込みオープン

```
      SUBROUTINE GFAOPN
     M         ( IFILE ,
     I           HFILE  )
*
      INTEGER    IFILE                    !" 装置番号
      CHARACTER  HFILE*(*)                !" ファイル名
```

4.2.6 GFOOPN [S] ファイル (追加) 書き込みオープン

```
      SUBROUTINE GFOOPN
     M         ( IFILE ,
     I           HFILX , OAPPND  )
*
      INTEGER    IFILE                    !" 装置番号
      CHARACTER  HFILX*(*)                !" ファイル名
      LOGICAL    OAPPND                   !" 追加?
```

4.2.7 GFROPQ [S] ファイル名取得& 読み込みオープン

```
      SUBROUTINE GFROPQ
     O         ( IFILE ,
     I           HINQ   )
```

4.2.8 GFWOPQ [ES] ファイル名取得& 書き込みオープン

```
      ENTRY      GFWOPQ
     O         ( IFILE ,
     I           HINQ   )
*
      INTEGER    IFILE                    !" 装置番号
      CHARACTER  HINQ  *(*)
```

4.2.9 GFAOPQ [ES] ファイル名取得& 追加書き込みオープン

```
      ENTRY      GFAOPQ
     O         ( IFILE ,
     I           HINQ   )
```

```
*
      INTEGER    IFILE                    !" 装置番号
      CHARACTER  HINQ  *(*)
```

4.2.10　GFOOPQ [ES] ファイル名取得& (追加) 書き込みオープン

```
       ENTRY      GFOOPQ
      M         ( IFILE ,
      I           HINQ    )
   *
       INTEGER    IFILE                     !" 装置番号
       CHARACTER  HINQ  *(*)
```

4.3　そのほかのモジュール

4.3.1　GFSADJ [ES] 整合寸法の設定

```
      ENTRY      GFSADJ
     I         ( IXDIM , IYDIM , IZDIM )
*
      INTEGER    IXDIM                     ! 第 1 次元の整合寸法
      INTEGER    IYDIM                     ! 第 2 次元の整合寸法
      INTEGER    IZDIM                     ! 第 3 次元の整合寸法
```

GTOOL3 では，通常はデータを配列に隙間なく入れて処理を行なう． 従って，モジュール GFREAD (4.1.1)，モジュール GFRDSL (4.1.2) はデータを配列に隙間なく詰め込み，モジュール GFWRIT (4.1.4) は隙間なく詰め込んであるものとして扱う． GTOOL3 を用いていないプログラムでは，必ずしもそのような形式になっているとは限らない． そのような場合など，配列の一部に入っているデータを GTOOL3 形式で入出力したい場合に，データの整合寸法をこのモジュールによって指定する．

IXDIM，IYDIM，IZDIM が 0 以外の時には，データが，

```
REAL    DATA ( IXDIM, IYDIM, IZDIM )
```

というように宣言されており，各々その始めの（添字 1 からデータの寸法までの）部分にデータが入っているものとして処理を行なう． もとに戻すには全て 0 を指定する．

例えば，大循環モデルプログラム GCM5 の T42 版においては第 1 次元方向に 128 個の格子点があるが，バンクコンフリクトを避けるなどの目的から配列の第 1 次元の寸法を 130 にとっている． このような場合，

```
CALL GFSADJ ( 130, 0   , 0   )
```

ととることによって，配列の第 1 次元の寸法が 130 であり，その始めの部分にデータが入っているとGFREAD，GFRDSL，GFWRITに知らせることができ，データを書き換えずにそのまま読み書きができる． この場合のように第 2 次元，第 3 次元方向には通常どおり隙間なく入っている場合は 0 を指定しておけばよい．

この整合寸法の指定が有効なのはモジュール GFREAD，GFRDSL，GFWRIT だけであり，それ以外のモジュールには通用しないので注意されたい．

現在の設定状態は次の GFQADJ によって参照できる．

4.3.2　GFQADJ [S] 整合寸法の参照

```
      SUBROUTINE GFQADJ
     I         ( IXDIM , IYDIM , IZDIM )
*
      INTEGER    IXDIM                        ! 第 1 次元の整合寸法
      INTEGER    IYDIM                        ! 第 2 次元の整合寸法
      INTEGER    IZDIM                        ! 第 3 次元の整合寸法
```

モジュール GFSADJ (4.3.1) 参照．

4.3.3　GFWTTS [ES] GTOOL3 標準フォーマット時間設定出力

```
      ENTRY      GFWTTS
     I         ( HHEAD , GDATA ,
     I           IT    , IDATE , IDNUM , ITDUR ,
     I           JFILE , IOMODE, NOEND           )
*
      CHARACTER  HHEAD ( * )*(*)             !" ヘッダー
      REAL       GDATA ( * )                 !" データ
      INTEGER    IEOD                        !" データ有：0　無：1
      INTEGER    IT                          !" 通し時間
      INTEGER    IDATE ( * )                 !" 時刻：年月日時分秒
      INTEGER    IDNUM                       !" データ番号
      INTEGER    ITDUR                       !" 代表時間
      INTEGER    JFILE                       !" 出力ファイル番号
      INTEGER    IOMODE                      !" 入出力モード
      INTEGER    NOEND                       !" ENDFILE 間隔
```

4.3.4　GFRDTS [S] GTOOL3 標準フォーマット入力時間取得

```
      SUBROUTINE GFRDTS
     O         ( HHEAD , GDATA , IEOD  ,
     O           IT    , IDATE , IDNUM , ITDUR ,
     I           HPSEL , HSEL  , ISEL  ,
     I           IFILE , IOMODE               )
```

```
*
      CHARACTER  HHEAD ( * )*(*)          !" ヘッダー
      REAL       GDATA ( * )              !" データ
      INTEGER    IEOD                     !" データ有：0　無：1
      INTEGER    IT                       !" 通し時間
      INTEGER    IDATE ( * )              !" 時刻：年月日時分秒
      INTEGER    IDNUM                    !" データ番号
      INTEGER    ITDUR                    !" 代表時間
      CHARACTER  HPSEL *(*)               !" 選択する欄
      CHARACTER  HSEL  *(*)               !" 選択するデータ (文
*    (INTEGER)   ISEL                     !" 選択するデータ (数
      INTEGER    IFILE                    !" 入力ファイル番号
      INTEGER    IOMODE                   !" 入出力モード
      INTEGER    NOEND                    !" ENDFILE 間隔
```

第5章　データ加工モジュール群

データをさまざまに加工するモジュールが用意されている. ヘッダーは, データの変更に応じて自動的に変更される.

5.1　各軸に対する操作のモジュール

名称が GPX，GPY，GPZ で始まるモジュールは, それぞれ第 1 軸, 第 2 軸, 第 3 軸に対する各種操作を行なう. 以下には, GPX で始まる第 1 軸に対する操作のモジュールのみを示す. 第 2 軸あるいは第 3 軸に対する操作のモジュールは, それぞれ X を Y あるいは Z に, 第 1 軸を第 2 軸あるいは第 3 軸に読み変えればよい.

5.1.1　GPXSEL [S] 第 1 次元の格子選択

```
      SUBROUTINE GPXSEL
     I         ( HHEAD , GDATA , IXSEL  ,
     I           HEDIT , HETTL ,
     O           HHEADO, GDATAO          )
*
      CHARACTER  HHEAD  ( * )*(*)        ! ヘッダー (入力)
      REAL       GDATA  ( * )            ! データ (入力)
      INTEGER    IXSEL                   ! 選択する第 1 次元の格子
      CHARACTER  HEDIT       *(*)        ! 編集略記号
      CHARACTER  HETTL       *(*)        ! 編集タイトル
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー (出力)
      REAL       GDATAO ( * )            ! データ (出力)
```

第 1 次元に関して，特定の格子 (格子番号 IXSEL) のデータのみを取り出す． 次元は一つ下がり，第 2 次元の情報が第 1 次元の情報に，第 3 次元の情報が第 2 次元の情報に順次移動される． 出力データは (入力データの第 2 次元の寸法)×(入力データの第 3 次元の寸法) の大きさである.

もともとのデータに存在しない格子を指定した場合には欠損値が代入される． ただし周期的な座標軸の場合には周期性が考慮される.

編集略記号として空白 (' ') を指定した場合には，編集略記号として'XS' に IXSEL の内容がついたもの (例えば'XS5') が，編集タイトルとして '(第 1 次元の格子の名称)=(格子の位置)' (例えば'GLON128=30.') が編集記述子として付加される． この際，格子位置情報が参照される． 編集略記号として'NUL' を指定した場合には，編集記述子は付加されない． そのほかの場合は，与えた HEDIT,HETTL がそのまま編集記述子として付加される.

5.1.2　GPXRDC [S] 第 1 次元の部分切出し

```
      SUBROUTINE GPXRDC
     I         ( HHEAD , GDATA , IXSEL1 , IXSEL2 ,
     I           HEDIT , HETTL ,
     O           HHEADO, GDATAO                    )
*
      CHARACTER  HHEAD  ( * )*(*)        ! ヘッダー (入力)
      REAL       GDATA  ( * )            ! データ (入力)
      INTEGER    IXSEL1                  ! 切り出す第 1 次元の格子
      INTEGER    IXSEL2                  ! 切り出す第 1 次元の格子
      CHARACTER  HEDIT       *(*)        ! 編集略記号
      CHARACTER  HETTL       *(*)        ! 編集タイトル
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー (出力)
      REAL       GDATAO ( * )            ! データ (出力)
```

第 1 次元に関して，特定の部分（格子番号 IXSEL1 から IXSEL2 の部分）のデータを取り出す．次元は下がらない． IXSEL1 と IXSEL2 に同じ数を指定しても，ヘッダーは GPXSEL の結果と同じにならないので注意． 出力データは (IXSEL2-IXSEL1+1)×（入力データの第 2 次元の寸法)×(入力データの第 3 次元の寸法）の大きさである．

もともとのデータに存在しない領域には欠損値が代入される． ただし周期的な座標軸の場合には周期性が考慮される．

編集略記号として空白 (' ') を指定した場合には，編集略記号として'XR' に IXSEL1 と IXSEL2 の内容がついたもの (例えば'XR5-10') が，編集タイトルとして '(第 1 次元の格子の名称)=(格子の位置 1)-(格子の位置 2)' (例えば'GLON128=30.-60.') が編集記述子として付加される．この際，格子位置情報が参照される． 編集略記号として'NUL' を指定した場合には，編集記述子は付加されない． そのほかの場合は，与えた HEDIT,HETTL がそのまま編集記述子として付加される．

5.1.3　GPXAVG [S] 第 1 次元に関する平均

```
      SUBROUTINE GPXAVG
     I         ( HHEAD , GDATA ,
     I           HEDIT , HETTL ,
     O           HHEADO, GDATAO  )
*
      CHARACTER  HHEAD  ( * )*(*)        ! ヘッダー (入力)
      REAL       GDATA  ( * )            ! データ (入力)
      CHARACTER  HEDIT       *(*)        ! 編集略記号
      CHARACTER  HETTL       *(*)        ! 編集タイトル
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー (出力)
      REAL       GDATAO ( * )            ! データ (出力)
```

データの第 1 次元に関する平均を作成する． 格子重みファイルが参照され，それに記述された重みをかけた平均をとる． 出力データは (入力データの第 2 次元の寸法)×(入力データの第 3 次元の寸法) の大きさである．

編集略記号として空白 (' ') を指定した場合には，編集略記号として'XM' が，編集タイトルとして '(第 1 次元の格子の名称)-mean' (例えば'GLON128-mean') が編集記述子として付加される． 編集略記号として'NUL' を指定した場合には，編集記述子は付加されない． そのほかの場合は，与えた HEDIT,HETTL がそのまま編集記述子として付加される．

5.1.4　GPXEDY [S] 第 1 次元の平均を引く

```
      SUBROUTINE GPXEDY
     I         ( HHEAD , GDATA ,
     I           HEDIT , HETTL ,
     O           HHEADO, GDATAO )
*
      CHARACTER  HHEAD  ( * )*(*)        ! ヘッダー (入力)
      REAL       GDATA  ( * )            ! データ (入力)
      CHARACTER  HEDIT       *(*)        ! 編集略記号
      CHARACTER  HETTL       *(*)        ! 編集タイトル
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー (出力)
      REAL       GDATAO ( * )            ! データ (出力)
```

データの第 1 次元に関する平均を，データから差し引いたものを作成する． 出力データは入力データと同じ大きさである． 平均を入れておくための作業領域 GWORK ((入力データの第 2 次元の寸法)×(入力データの第 3 次元の寸法) の大きさ) が必要である．

編集略記号として空白 (' ') を指定した場合には，編集略記号として'XE' が，編集タイトルとして '(第 1 次元の格子の名称)-eddy' (例えば'GLON128-eddy') が編集記述子として付加される． 編集略記号として'NUL' を指定した場合には，編集記述子は付加されない． そのほかの場合は，与えた HEDIT,HETTL がそのまま編集記述子として付加される．

5.1.5　GPXCOM [S] 第 1 次元の結合

```
      SUBROUTINE GPXCOM
     I         ( HHEADO, GDATA , IX    ,
     M           GDATAO                )
*
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー (出力)
      REAL       GDATA  ( * )            ! データ (入力)
      INTEGER    IX                      ! 入れるべき第 1 次元の格
      REAL       GDATAO ( * )            ! データ (出力)
```

1 次元のデータを集めて 2 次元のデータを，あるいは 2 次元のデータを集めて 3 次元のデータを作成する． 入力データを，第 1 次元の格子番号 IX の格子のデータとして出力データに入れる．

次元は一つ上がり，第 2 次元の情報が第 3 次元の情報に，第 1 次元の情報が第 2 次元の情報に順次移動される．

これを用いるためには，あらかじめ次にあげるモジュール GPXCMI を呼んでおく必要がある．すなわち，GPXCMI によって出力領域を確保し，ヘッダーををセットした後，GPXCOM を順次呼んでデータを入れて行く．　入れて行くデータは全て同じ形式である必要がある．

5.1.6　GPXCMI [S] 第 1 次元の結合の準備

```
      SUBROUTINE GPXCMI
     I         ( HHEAD , HAITMX, IXSTR , IXEND ,
     I           HEDIT , HETTL ,
     O           HHEADO, GDATAO )
*
      CHARACTER  HHEAD  ( * )*(*)        ! ヘッダー (入力)
      CHARACTER  HAITMX      *(*)        ! 結合する第 1 次元の格子
      INTEGER    IXSTR                   ! 第 1 次元の軸の格子始め
      INTEGER    IXEND                   ! 第 1 次元の軸の格子終り
      CHARACTER  HEDIT       *(*)        ! 編集略記号
      CHARACTER  HETTL       *(*)        ! 編集タイトル
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー (出力)
      REAL       GDATAO ( * )            ! データ (出力)
```

GPXCOM によってデータを結合する準備をする．　格子番号 IXSTR から IXEND までの出力領域を確保し，ヘッダーを設定する．　出力データは (IXEND-IXSTR+1)×（入力データの第 2 次元の寸法)×(入力データの第 3 次元の寸法）の大きさである．

IXSTR=IXEND=0 の時には，格子位置ファイルを参照し，1 から全格子数までの出力領域を確保する．

編集略記号として空白 (' ') を指定した場合には，編集略記号として'XC' に IXSTR と IXEND の内容がついたもの (例えば'XC5-10') が，編集タイトルとして '(第 1 次元の格子の名称)=(格子の位置 1)-(格子の位置 2)' (例えば'GLON128=30.-60.') が編集記述子として付加される．この際，格子位置ファイルが参照される．　編集略記号として'NUL' を指定した場合には，編集記述子は付加されない．　そのほかの場合は，与えた HEDIT,HETTL がそのまま編集記述子として付加される．

5.1.7　GPXEXP [S] 第 1 次元を拡張

```
      SUBROUTINE GPXEXP
     I         ( HHEAD , GDATA ,
     I           HEDIT , HETTL ,
     O           HHEADO, GDATAO   )
*
      CHARACTER  HHEAD  ( * )*(*)        ! ヘッダー (入力)
      REAL       GDATA  ( * )            ! データ (入力)
      CHARACTER  HEDIT       *(*)        ! 編集略記号
```

```
      CHARACTER  HETTL       *(*)        ! 編集タイトル
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー (出力)
      REAL       GDATAO ( * )            ! データ (出力)
```

第 1 次元を，格子点位置ファイルの情報に基づいて 1 から全格子数までの領域に拡張する． もともとのデータに存在しない領域には欠損値が代入される． 出力データは (第 1 次元の全格子数)×（入力データの第 2 次元の寸法)×(入力データの第 3 次元の寸法）の大きさである．

内部でモジュール GPXRDC (5.1.2) を呼び出しており，編集記述の設定等は GDXRDC に準ずる．

5.1.8　GPXEXC [ES] 第 1 次元をサイクリックに拡張

```
      ENTRY      GPXEXC
     I         ( HHEAD , GDATA ,
     I           HEDIT , HETTL ,
     O           HHEADO, GDATAO   )
*
      CHARACTER  HHEAD  ( * )*(*)        ! ヘッダー (入力)
      REAL       GDATA  ( * )            ! データ (入力)
      CHARACTER  HEDIT       *(*)        ! 編集略記号
      CHARACTER  HETTL       *(*)        ! 編集タイトル
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー (出力)
      REAL       GDATAO ( * )            ! データ (出力)
```

第 1 次元を，格子点位置ファイルの情報に基づいて 1 から全格子数までの領域に拡張する． もともとのデータに存在しない領域には欠損値が代入される． ただし周期的な座標軸の場合には，(全格子数+1) までの領域に周期性を考慮して拡張される． 非周期的な座標軸の場合は，モジュール GPXEXP (5.1.7) と同じ結果となる． 出力データは非周期的な座標軸の場合 (第 1 次元の全格子数)×（入力データの第 2 次元の寸法)×(入力データの第 3 次元の寸法）の大きさ，非周期的な座標軸の場合 (第 1 次元の全格子数+1)×（入力データの第 2 次元の寸法)×(入力データの第 3 次元の寸法）の大きさである．

内部でモジュール GPXRDC (5.1.2) を呼び出しており，編集記述の設定等は GDXRDC に準ずる．

5.1.9　GPXEXT [S] 第 1 次元の拡大

```
      SUBROUTINE GPXEXT
     I         ( HHEAD , GDATA ,
     I           HAITMX, IXSTR , IXEND ,
     I           HEDIT , HETTL ,
     O           HHEADO, GDATAO )
*
      CHARACTER  HHEAD  ( * )*(*)        !" ヘッダー (入力)
      REAL       GDATA  ( * )            !" データ (入力)
      CHARACTER  HAITMX      *(*)        !" 拡大する第 1 次元の軸名
```

```
      INTEGER    IXSTR                  !" 第 1 次元の軸の格子始め
      INTEGER    IXEND                  !" 第 1 次元の軸の格子終り
      CHARACTER  HEDIT       *(*)       !" 編集略記号
      CHARACTER  HETTL       *(*)       !" 編集タイトル
      CHARACTER  HHEADO ( * )*(*)       !" ヘッダー (出力)
      REAL       GDATAO ( * )           !" データ (出力)
```

5.2　時系列に関する操作のモジュール

時刻ごとに別のデータになっている時系列データを処理するモジュールである.

5.2.1　GPTAVG [S] 時間平均への足し込み

```
      SUBROUTINE GPTAVG
     I         ( HHEAD , GDATA ,
     I           HEDIT , HETTL ,
     M           HHEADW, WDATA ,
     O           HHEADO, GDATAO )
*
      CHARACTER  HHEAD  ( * )*(*)       ! ヘッダー (入力)
      REAL       GDATA  ( * )           ! データ (入力)
      CHARACTER  HEDIT       *(*)       ! 編集略記号
      CHARACTER  HETTL       *(*)       ! 編集タイトル
      CHARACTER  HHEADW ( * )*(*)       ! ヘッダー (ワーク)
      REAL       WDATA  ( * )           ! ウェイトカウンター
      CHARACTER  HHEADO ( * )*(*)       ! ヘッダー (出力)
      REAL       GDATAO ( * )           ! データ (出力)
```

時系列データの時間に対する平均を作成する． GPTAVG，GPTAVO，GPTAVR の 3 つのモジュールを一組で使用する． まず，GPTAVR で初期化を行なう． その後 GPTAVG によってデータを順次入れて行く． 最後に GPTAVO を呼ぶことによって時間平均が求められる． 作業領域として，入力データと同じ大きさの WDATA が必要である． 出力データは入力データと同じ大きさである.

時間平均の重みとしては，ヘッダーの欄 TDUR(28)，すなわちデータの代表する時間が使用される.

編集略記号として空白 (' ') を指定した場合には，編集略記号として'TM' が，編集タイトルとして最初のデータの時刻を，モジュール GTPGET/GTPSET (2.1.1) の管理するパラメータ UTIM (既定値は DAY) の単位で示したもの (例えば'120.DAY-') が編集記述子として付加される. 編集略記号として'NUL' を指定した場合には，編集記述子は付加されない． そのほかの場合は，与えた HEDIT,HETTL がそのまま編集記述子として付加される.

平均結果の時刻は最後のデータの時刻となり，データ代表時間 (欄 TDUR(28)) には，入力データのデータ代表時間の和が入れられる.

5.2.2　GPTAVO [S] 時間平均の算出

```
      SUBROUTINE GPTAVO
     I         ( HHEADW, WDATA  ,
     M           HHEADO, GDATAO  )
*
      CHARACTER  HHEADW ( * )*(*)        ! ヘッダー (ワーク)
      REAL       WDATA  ( * )            ! ウェイトカウンター
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー (出力)
      REAL       GDATAO ( * )            ! データ (出力)
```

時間平均の算出を行なう．くわしくはモジュール GPTAVG (5.2.1) の項参照．

5.2.3　GPTAVR [ES] 時間平均のリセット

```
      ENTRY       GPTAVR
     O          ( HHEADO )
*
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー (出力)
```

時間平均の初期化を行なう．実は HHEADO に初期化の印を付けているだけであり，実際上の初期化処理は GPTAVG の中で行なう．くわしくはモジュール GPTAVG (5.2.1) の項を参照．

5.2.4　GPTSEQ [S] 時系列の作成

```
      SUBROUTINE  GPTSEQ
     I          ( HHEAD , GDATA ,
     I            HEDIT , HETTL ,
     O            HHEADT, GDATAT )
*
      CHARACTER  HHEAD  ( * )*(*)        ! ヘッダー (入力)
      REAL       GDATA  ( * )            ! データ (入力)
      CHARACTER  HEDIT       *(*)        ! 編集略記号
      CHARACTER  HETTL       *(*)        ! 編集タイトル
      CHARACTER  HHEADT ( * )*(*)        ! ヘッダー (出力)
      REAL       GDATAT ( * )            ! データ (出力)
```

時刻ごとに別のデータとなっているデータから，一つの座標軸が時間軸であるような時系列データを作成する．GPTSEQ，GPTSQR の 2 つのモジュールを一組で使用する．まず，GPTSQR で初期化を行なう．その後 GPTSEQ によってデータを順次入れて行く．

入力データが 1 次元データの場合は 2 次元めが時間軸となり，入力データが 2 次元データの場合は 3 次元めが時間軸となる．

内部で'@TIMExx' という名称の格子 (xx は数字であり，新しい時系列を作成しはじめるたびに 1 から増えて行く) を生成し，モジュール GUQIAX/GUSIAX (7.3.4) に管理させる．この軸

の単位は，モジュール GTPGET/GTPSET (2.1.1) の管理するパラメータ UTIM (既定値は 'DAY') である．

編集略記号として空白 (' ') を指定した場合には，編集略記号として'TSEQ'が，編集タイトルとして'time sequence'が編集記述子として付加される． 編集略記号として'NUL'を指定した場合には，編集記述子は付加されない． そのほかの場合は，与えた HEDIT,HETTL がそのまま編集記述子として付加される．

5.2.5　GPTSQR [ES] 時系列のリセット

```
      ENTRY       GPTSQR
     O          ( HHEADT )
*
      CHARACTER  HHEADT ( * )*(*)        ! ヘッダー (出力)
```

時系列作成の初期化を行なう． 実は HHEADT に初期化の印を付けているだけであり，実際上の初期化処理は GPTSEQ の中で行なう． くわしくはモジュール GPTSEQ (5.2.4) 参照．

5.2.6　GPTSQN [ES] 時系列番号のセット

```
      ENTRY       GPTSQN
     I          ( ISEQ   )
```

5.3　1 組のデータに対する基本演算操作のモジュール

以下は, 電脳ライブラリ MATH1 における VRALIB に対応したモジュールである.

5.3.1　GPFINC [S] データとスカラーの和

```
      SUBROUTINE GPFINC
     I         ( HHEAD , GDATA , VAL   ,
     I           HEDIT , HETTL ,
     O           HHEADO, GDATAO  )
*
      CHARACTER  HHEAD  ( * )*(*)        ! ヘッダー (入力)
      REAL       GDATA  ( * )            ! データ (入力)
      REAL       VAL                     ! スカラー値
      CHARACTER  HEDIT       *(*)        ! 編集略記号
      CHARACTER  HETTL       *(*)        ! 編集タイトル
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー (出力)
      REAL       GDATAO ( * )            ! データ (出力)
```

配列データ GDATA の各々の要素とスカラー VAL を足し合わせ，配列データ GDATAO として格納する． SUBFUNC の VRINC に対応する．

編集略記号として空白（’ ’）または’NUL’を指定した場合には，編集記述子は付加されない．そのほかの場合は，与えた HEDIT,HETTL がそのまま編集記述子として付加される．

5.3.2　GPFFCT [S] データとスカラーの積

```
      SUBROUTINE GPFFCT
     I         ( HHEAD , GDATA , VAL   ,
     I           HEDIT , HETTL ,
     O           HHEADO, GDATAO  )
*
      CHARACTER  HHEAD  ( * )*(*)        ! ヘッダー（入力）
      REAL       GDATA  ( * )            ! データ（入力）
      REAL       VAL                     ! スカラー値
      CHARACTER  HEDIT       *(*)        ! 編集略記号
      CHARACTER  HETTL       *(*)        ! 編集タイトル
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー（出力）
      REAL       GDATAO ( * )            ! データ（出力）
```

配列データ GDATA の各々の要素とスカラー VAL をかけ合わせ，配列データ GDATAO として格納する． SUBFUNC の VRFCT に対応する． その他はモジュール GPFINC (5.3.1) を参照．

5.3.3　GPFCON [S] データへのスカラー代入

```
      SUBROUTINE GPFCON
     I         ( HHEAD , GDATA , VAL   ,
     I           HEDIT , HETTL ,
     O           HHEADO, GDATAO  )
*
      CHARACTER  HHEAD  ( * )*(*)        ! ヘッダー（入力）
      REAL       GDATA  ( * )            ! データ（入力）
      REAL       VAL                     ! スカラー値
      CHARACTER  HEDIT       *(*)        ! 編集略記号
      CHARACTER  HETTL       *(*)        ! 編集タイトル
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー（出力）
      REAL       GDATAO ( * )            ! データ（出力）
```

配列データ GDATAO の各々の要素にスカラー VAL を代入する． ただし，配列データ GDATA の内容が欠損値である要素には欠損値が入る． SUBFUNC の VRCON に対応する． その他はモジュール GPFINC (5.3.1) を参照．

5.3.4　GPFSET [S] データにデータを代入

```
      SUBROUTINE GPFSET
```

```
      I         ( HHEAD , GDATA ,
      I           HEDIT , HETTL ,
      O           HHEADO, GDATAO  )
*
       CHARACTER  HHEAD  ( * )*(*)        ! ヘッダー (入力)
       REAL       GDATA  ( * )            ! データ (入力)
       CHARACTER  HEDIT       *(*)        ! 編集略記号
       CHARACTER  HETTL       *(*)        ! 編集タイトル
       CHARACTER  HHEADO ( * )*(*)        ! ヘッダー (出力)
       REAL       GDATAO ( * )            ! データ (出力)
```

配列データ GDATA を配列データ GDATAO に代入する． SUBFUNC の VRSET に対応する． その他はモジュール GPFINC (5.3.1) を参照．

5.3.5　GPFFNA [S] データに関数を作用

```
       SUBROUTINE GPFFNA
      I         ( HHEAD , GDATA , RFNA  ,
      I           HEDIT , HETTL ,
      O           HHEADO, GDATAO  )
*
       CHARACTER  HHEAD  ( * )*(*)        ! ヘッダー (入力)
       REAL       GDATA  ( * )            ! データ (入力)
       EXTERNAL   RFNA                    ! 関数 (1 変数)
       CHARACTER  HEDIT       *(*)        ! 編集略記号
       CHARACTER  HETTL       *(*)        ! 編集タイトル
       CHARACTER  HHEADO ( * )*(*)        ! ヘッダー (出力)
       REAL       GDATAO ( * )            ! データ (出力)
```

配列データ GDATA の各々の要素に関数 FNA を作用させ配列データ GDATAO として格納する． SUBFUNC の VRFNA に対応する． その他はモジュール GPFINC (5.3.1) を参照．

5.4　2 組のデータに対する基本演算操作のモジュール

以下は, 電脳ライブラリ MATH1 における VRBLIB に対応したモジュールである.

5.4.1　GPFADD [S] データ同士の和

```
       SUBROUTINE GPFADD
      I         ( HHEAD1, GDATA1,
      I           HHEAD2, GDATA2,
      I           HEDIT , HETTL ,
      O           HHEADO, GDATAO  )
*
```

```
      CHARACTER  HHEAD1 ( * )*(*)        ! ヘッダー (入力 1)
      REAL       GDATA1 ( * )            ! データ (入力 1)
      CHARACTER  HHEAD2 ( * )*(*)        ! ヘッダー (入力 2)
      REAL       GDATA2 ( * )            ! データ (入力 2)
      CHARACTER  HEDIT       *(*)        ! 編集略記号
      CHARACTER  HETTL       *(*)        ! 編集タイトル
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー (出力)
      REAL       GDATAO ( * )            ! データ (出力)
```

配列データ GDATA1 と GDATA2 を要素ごとに足し合わせ，配列データ GDATAO として格納する．SUBFUNC の VRADD に対応する．

出力配列データ GDATAO の範囲は GDATA1 の領域と同じとなる． GDATA1 と GDATA2 の範囲は一致していなくてよいが，各軸の格子の名称が一致していない時には警告が出る．

編集略記号として空白 (' ') または'NUL'を指定した場合には，編集記述子は付加されない．そのほかの場合は，与えた HEDIT,HETTL がそのまま編集記述子として付加される．

5.4.2　GPFSUB [S] データ同士の差

```
      SUBROUTINE GPFSUB
     I         ( HHEAD1, GDATA1,
     I           HHEAD2, GDATA2,
     I           HEDIT , HETTL ,
     O           HHEADO, GDATAO  )
*
      CHARACTER  HHEAD1 ( * )*(*)        ! ヘッダー (入力 1)
      REAL       GDATA1 ( * )            ! データ (入力 1)
      CHARACTER  HHEAD2 ( * )*(*)        ! ヘッダー (入力 2)
      REAL       GDATA2 ( * )            ! データ (入力 2)
      CHARACTER  HEDIT       *(*)        ! 編集略記号
      CHARACTER  HETTL       *(*)        ! 編集タイトル
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー (出力)
      REAL       GDATAO ( * )            ! データ (出力)
```

配列データ GDATA1 の各要素から GDATA2 の各要素を引き配列データ GDATAO として格納する．SUBFUNC の VRSUB に対応する． その他はモジュール GPFADD (5.4.1) を参照．

5.4.3　GPFMLT [S] データ同士の積

```
      SUBROUTINE GPFMLT
     I         ( HHEAD1, GDATA1,
     I           HHEAD2, GDATA2,
     I           HEDIT , HETTL ,
     O           HHEADO, GDATAO  )
```

```
*
      CHARACTER  HHEAD1 ( * )*(*)        ! ヘッダー (入力 1)
      REAL       GDATA1 ( * )            ! データ (入力 1)
      CHARACTER  HHEAD2 ( * )*(*)        ! ヘッダー (入力 2)
      REAL       GDATA2 ( * )            ! データ (入力 2)
      CHARACTER  HEDIT       *(*)        ! 編集略記号
      CHARACTER  HETTL       *(*)        ! 編集タイトル
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー (出力)
      REAL       GDATAO ( * )            ! データ (出力)
```

配列データ GDATA1 と GDATA2 を要素ごとにかけ合わせ，配列データ GDATAO として格納する．
SUBFUNC の VRMUL に対応する． その他はモジュール GPFADD (5.4.1) を参照．

5.4.4 GPFDIV [S] データ同士の商

```
      SUBROUTINE GPFDIV
     I         ( HHEAD1, GDATA1,
     I           HHEAD2, GDATA2,
     I           HEDIT , HETTL ,
     O           HHEADO, GDATAO  )
*
      CHARACTER  HHEAD1 ( * )*(*)        ! ヘッダー (入力 1)
      REAL       GDATA1 ( * )            ! データ (入力 1)
      CHARACTER  HHEAD2 ( * )*(*)        ! ヘッダー (入力 2)
      REAL       GDATA2 ( * )            ! データ (入力 2)
      CHARACTER  HEDIT       *(*)        ! 編集略記号
      CHARACTER  HETTL       *(*)        ! 編集タイトル
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー (出力)
      REAL       GDATAO ( * )            ! データ (出力)
```

配列データ GDATA1 の各要素を GDATA2 の各要素で割り配列データ GDATAO として格納する．
SUBFUNC の VRDIV に対応する． その他はモジュール GPFADD (5.4.1) を参照．

5.4.5 GPFFNB [S] 2 つのデータに関数を作用

```
      SUBROUTINE GPFFNB
     I         ( HHEAD1, GDATA1,
     I           HHEAD2, GDATA2, RFNB  ,
     I           HEDIT , HETTL ,
     O           HHEADO, GDATAO  )
*
      CHARACTER  HHEAD1 ( * )*(*)        ! ヘッダー (入力 1)
      REAL       GDATA1 ( * )            ! データ (入力 1)
      CHARACTER  HHEAD2 ( * )*(*)        ! ヘッダー (入力 2)
```

```
      REAL       GDATA2 ( * )            ! データ (入力 2)
      EXTERNAL   RFNB                    ! 関数 (2 変数)
      CHARACTER  HEDIT       *(*)        ! 編集略記号
      CHARACTER  HETTL       *(*)        ! 編集タイトル
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー (出力)
      REAL       GDATAO ( * )            ! データ (出力)
```

配列データ GDATA と GDATA2 の各要素を 2 変数関数 FNB に入れた結果を，配列データ GDATAO として格納する． SUBFUNC の VRFNB に対応する． その他はモジュール GPFADD (5.4.1) を参照．

5.5　外部手続き呼びだし汎用モジュール

5.5.1　GPCAL1 [S] 外部手続き呼びだし

```
      SUBROUTINE GPCAL1
     I         ( GESUBR,
     I           HHEAD , GDATA ,
     I           HEDIT , HETTL ,
     O           HHEADO, GDATAO )
*
*     EXTERNAL   GESUBR                  ! 外部手続き名
      CHARACTER  HHEAD  ( * )*(*)        ! ヘッダー (入力)
      REAL       GDATA  ( * )            ! データ (入力)
      CHARACTER  HEDIT       *(*)        ! 編集略記号
      CHARACTER  HETTL       *(*)        ! 編集タイトル
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー (出力)
      REAL       GDATAO ( * )            ! データ (出力)
```

次のようなユーザー作成の外部手続き (サブルーチン) を呼び出す．

```
       SUBROUTINE NANASI
      I         ( HHEAD , GDATA ,
      O           HHEADO, GDATAO,
      D           IXDIM , IYDIM , IZDIM )
    *
       REAL       GDATA ( IXDIM, IYDIM, IZDIM )
```

GDATAO の呼ばれる側でのサイズは自由である． GPEXS1 の中では GDATAO のサイズのチェックは行なわれないので，必要ならば呼ばれる側のサブルーチンで GUSZCK を用いてチェックする必要がある． また，HHEADO は HHEAD のコピーが渡され，GPEXS1 の中では操作が行なわれないので，必要ならば呼ばれる側のサブルーチンで適当にセットし直す必要がある．

編集略記号として空白(' ') または'NUL'以外のものを指定した場合には，与えたHEDIT,HETTLがそのまま編集記述子として付加される． 呼ばれる側のサブルーチンで編集記述子を付加した場合にはそれに上書きされる．

5.5.2　GPCAL2 [S] 2引数外部手続き呼びだし

```
      SUBROUTINE GPCAL2
     I         ( GESUBR,
     I           HHEAD1, GDATA1,
     I           HHEAD2, GDATA2,
     I           HEDIT , HETTL ,
     O           HHEADO, GDATAO )
*
*     EXTERNAL   GESUBR                  ! 外部手続き名
      CHARACTER  HHEAD1 ( * )*(*)        ! ヘッダー(入力 1)
      REAL       GDATA1 ( * )            ! データ(入力 1)
      CHARACTER  HHEAD2 ( * )*(*)        ! ヘッダー(入力 2)
      REAL       GDATA2 ( * )            ! データ(入力 2)
      CHARACTER  HEDIT       *(*)        ! 編集略記号
      CHARACTER  HETTL       *(*)        ! 編集タイトル
      CHARACTER  HHEADO ( * )*(*)        ! ヘッダー(出力)
      REAL       GDATAO ( * )            ! データ(出力)
```

次のようなユーザー作成の外部手続き(サブルーチン)を呼び出す．

```
       SUBROUTINE NANAS2
      I         ( HHEAD1, GDATA1,
      I           HHEAD2, GDATA2,
      O           HHEADO, GDATAO,
      D           IXDIM1, IYDIM1, IZDIM1,
      D           IXDIM2, IYDIM2, IZDIM2 )
   *
       REAL       GDATA1( IXDIM1, IYDIM1, IZDIM1 )
       REAL       GDATA2( IXDIM2, IYDIM2, IZDIM2 )
```

GDATAOの呼ばれる側でのサイズは自由である． HHEADOはHHEAD1のコピーが渡される． そのほかはモジュール GPCAL1 (5.5.1) と同様である．

5.5.3　GPCAL3 [S] 3引数外部手続き呼びだし

```
      SUBROUTINE GPCAL3
     I         ( GESUBR,
     I           HHEAD1, GDATA1,
     I           HHEAD2, GDATA2,
```

```
      I           HHEAD3, GDATA3,
      I           HEDIT , HETTL ,
      O           HHEADO, GDATAO )
*
*     EXTERNAL    GESUBR                ! 外部手続き名
      CHARACTER   HHEAD1 ( * )*(*)      ! ヘッダー (入力 1)
      REAL        GDATA1 ( * )          ! データ (入力 1)
      CHARACTER   HHEAD2 ( * )*(*)      ! ヘッダー (入力 2)
      REAL        GDATA2 ( * )          ! データ (入力 2)
      CHARACTER   HHEAD3 ( * )*(*)      ! ヘッダー (入力 3)
      REAL        GDATA3 ( * )          ! データ (入力 3)
      CHARACTER   HEDIT       *(*)      ! 編集略記号
      CHARACTER   HETTL       *(*)      ! 編集タイトル
      CHARACTER   HHEADO ( * )*(*)      ! ヘッダー (出力)
      REAL        GDATAO ( * )          ! データ (出力)
```

次のようなユーザー作成の外部手続き (サブルーチン) を呼び出す.

```
       SUBROUTINE NANAS3
      I         ( HHEAD1, GDATA1,
      I           HHEAD2, GDATA2,
      I           HHEAD3, GDATA3,
      O           HHEADO, GDATAO,
      D           IXDIM1, IYDIM1, IZDIM1,
      D           IXDIM2, IYDIM2, IZDIM2,
      D           IXDIM3, IYDIM3, IZDIM3 )
*
       REAL       GDATA1( IXDIM1, IYDIM1, IZDIM1 )
       REAL       GDATA2( IXDIM2, IYDIM2, IZDIM2 )
       REAL       GDATA3( IXDIM3, IYDIM3, IZDIM3 )
```

GDATAO の呼ばれる側でのサイズは自由である. HHEADO は HHEAD1 のコピーが渡される. そのほかはモジュール GPCAL1 (5.5.1) と同様である.

5.5.4　GPCAL4 [S] 4 引数外部手続き呼びだし

```
       SUBROUTINE GPCAL4
      I         ( GESUBR,
      I           HHEAD1, GDATA1,
      I           HHEAD2, GDATA2,
      I           HHEAD3, GDATA3,
      I           HHEAD4, GDATA4,
      I           HEDIT , HETTL ,
      O           HHEADO, GDATAO )
```

```
*
*"    EXTERNAL   GESUBR                ! 外部手続き名
      CHARACTER  HHEAD1 ( * )*(*)      !" ヘッダー (入力 1)
      REAL       GDATA1 ( * )          !" データ (入力 1)
      CHARACTER  HHEAD2 ( * )*(*)      !" ヘッダー (入力 2)
      REAL       GDATA2 ( * )          !" データ (入力 2)
      CHARACTER  HHEAD3 ( * )*(*)      !" ヘッダー (入力 3)
      REAL       GDATA3 ( * )          !" データ (入力 3)
      CHARACTER  HHEAD4 ( * )*(*)      !" ヘッダー (入力 4)
      REAL       GDATA4 ( * )          !" データ (入力 4)
      CHARACTER  HEDIT       *(*)      !" 編集略記号
      CHARACTER  HETTL       *(*)      !" 編集タイトル
      CHARACTER  HHEADO ( * )*(*)      !" ヘッダー (出力)
      REAL       GDATAO ( * )          !" データ (出力)
```

第6章　描画モジュール群

地球流体電脳ライブラリを用いた, 描画に関するモジュールが用意されている.

6.1　基本描画モジュール

描画のためには, 各描画モジュールを次のような順番で呼ばなくてはならない.

```
GGOPEN
GGLAY1, GGLAY2, GGLAY3
GGAXES, GGAXIS
GGCNTR, GGTONE, GGVECT, GGCURV, GGMARK
GGCLSE
```

何枚も図を描く場合には, GGOPEN と GGCLSE は最初と最後に 1 回づつ呼ぶだけでよい. GGLAY1 etc. において改ページが行なわれる. GGCNTR などで描画した後も設定を記憶しているので, 例えばコンターとベクトルを重ねて表示することは, GGCNTR の直後に GGVECT を呼ぶことで実現できる. 折れ線を複数本描く場合も同様に GGCURV を続けて呼べばよい.
1 枚の図にコンターと折れ線を同居させるような場合等, レイアウトを変更したい場合には, GGLAY1 を書き直したものを作成して GGLAY1 の代わりに用いればよい.

6.1.1　GGOPEN [S] グラフィック開始

```
      SUBROUTINE GGOPEN
     I         ( IWS   )
*
      INTEGER    IWS                    ! ワークステーション No.
```

描画の初期化を行なう． 全ての他の描画モジュール呼び出しに先だって 1 回だけ呼ぶ必要がある． IWS については，SGPACK のマニュアルを参照のこと.

6.1.2　GGCLSE [S] グラフィック終了

```
      SUBROUTINE GGCLSE
```

描画の最終処理を行なう． 全ての他の描画モジュール呼び出しが終った後に 1 回だけ呼ぶ必要がある.

6.1.3　GGLAY1 [S] レイアウト, 題名表示

```
      SUBROUTINE GGLAY1
     I         ( HHEAD )
*
      CHARACTER  HHEAD ( * )*(*)          ! ヘッダー
```

枠のレイアウトを行ない，ヘッダーから必要な情報を取ってきて書き出す．

まずモジュール GGPGET/GGPSET (6.2.1) の管理するパラメータ VXMIN，VXMAX，VYMIN，VYMAX に従ってビューポートを設定する．

次にビューポートの上に，上からタイトル TITL1,TITL2 とデータセット名 DEST，単位 UNIT と時刻 TIME（モジュール GTPGET/GTPSET (2.1.1) の管理するパラメータ UITM (既定値は 'DAY') の単位に直したもの)，識別名称 ITEM と編集略記号の EDITx の並びとファイル番号 FNUM とデータ番号 DNUM を連ねたものを記す． 文字の高さはモジュール GGPGET/GGPSET (6.2.1) の管理するパラメータ STITL (既定値は 0.02) およびパラメータ SEDIT (既定値は 0.013) に従う．

さらに，ビューポートの右に上から順に編集タイトル ETITLx，データ時刻 DATE，横軸の格子識別名称 AITM1，横軸の格子範囲 ASTR1，AEND1，縦軸の格子識別名称 AITM2，縦軸の格子範囲 ASTR2，AEND2，レンジ最小値 DMIN，レンジ最大値 DMAX，間隔 (小)DIVS，間隔 (大)DIVL，データ変更日付 MDATE，データ変更者 MSIGN の順に記す． 文字の高さはモジュール GGPGET/GGPSET (6.2.1) の管理するパラメータ SDESC (既定値は 0.01) に従う．

他のレイアウト・題名表示サブルーチンとして GGLAY2,GGLAY3 が用意されている． サブルーチンの呼び方は GGLAY1 と同じである．

6.1.4　GGAXES [S] 軸のセット X-Y

```
      SUBROUTINE GGAXES
     I         ( HHEAD )
*
      CHARACTER  HHEAD ( * )*(*)          ! ヘッダー
```

コンター (GGCNTR)，トーン (GGTONE)，ベクトル (GGVECT) を描くための座標を設定し，座標軸を描く． これに先だってモジュール GGLAY1 (6.1.3) が呼ばれている必要がある．

まず格子位置ファイルを読み込み，USPACK を利用してウインドウを設定し，座標軸を描く．軸の範囲はヘッダーの記述子 ASTR1，ASTR2 に準拠し，さらにヘッダーの記述子 DMIN，DMAX，DIVS，DIVL を参考にする． これらに欠損値以外が設定されている場合には，DIVS ごとに目盛を打ち，DIVL ごとにラベルを付ける． しかも軸の全体を用いる場合には，DMIN から DMAX までのウィンドウを設定するようにする． それ以外のときは格子点の位置を用いて，USPACK によって適当に決める．

軸に付けるラベルの文字の高さ，および軸に付けるタイトルの文字の高さはモジュール GGPGET/GGPSET (6.2.1) の管理するパラメータ SAXSL (既定値は 0.015) およびパラメータ SAXST (既定値は 0.015) に従う．

軸の向き，および通常の軸か対数軸かは記述子 STYP(1.2.2 参照) に従う

6.1.5　GGAXIS [S] 軸のセット X or Y 軸のみ

```
      SUBROUTINE GGAXIS
     I         ( HHEAD , HPOS   )
*
      CHARACTER  HHEAD ( * )*(*)          ! ヘッダー
      CHARACTER  HPOS *(*)                ! セットする軸 ('X' or 'Y')
```

折れ線 (GGCURV)，マーカー (GGMARK) を描くための座標を設定し，座標軸を描く． これに先だってモジュール GGLAY1 (6.1.3) が呼ばれている必要がある．

HPOS='X' の時は横軸のみ，HPOS='Y' の時は縦軸のみ設定する． もう一つの軸の座標設定は，モジュール GGCURV (6.1.15)，モジュール GGMARK (6.1.16) の中で行なわれる．

そのほかはモジュール GGAXES (6.1.4) と同様である．

6.1.6　GGAXRR [ES] X or Y 軸のリセット

```
      ENTRY      GGAXRR
     I           ( HPOS )
      CHARACTER  HPOS *(*)                !" セットする軸 ('X' or 'Y')
```

6.1.7　GGCNTR [S] コンター描画

```
      SUBROUTINE GGCNTR
     I         ( HHEAD, GDATA )
*
      CHARACTER  HHEAD ( * )*(*)          ! ヘッダー
      REAL       GDATA ( * )              ! データ
```

UPACK の UDCNTR を用いてコンターを描画する． これに先だってモジュール GGAXES (6.1.4) が呼ばれている必要がある．

コンターレベルを決める際は，ヘッダーの記述子 DMIN，DMAX，DIVS，DIVL を参考にする． これらに欠損値以外が設定されている場合には，DMIN から DMAX まで，DIVS ごとにコンターを引き，DIVL ごとにラベルを付ける． それ以外のときは UDCNTR の内部で適当に決める． ただしこの場合，コンターに何本おきにラベルを付けるかの指定 ICYCLE はモジュール GGPGET/GGPSET (6.2.1) の管理するパラメータ ICYCLE (既定値は 4) に従う．

DIVS ごとにコンターを引いた場合にコンターの数がモジュール GGPGET/GGPSET (6.2.1) の管理するパラメータ NCONMAX (既定値は 50) 本以上になる場合には，パラメータ CONRFCT (既定値は 0.5) を参照してコンターレベルを設定し直す． CONRFCT=0. の時は大きな値のコンターを省き，CONRFCT=1. の時は小さな値のコンターを省く CONRFCT=0.5 の時は大きな値と小さな値を均等に省く． ただし，指定した DMIN が欠損値でなく，DMAX が欠損値の場合は CONRFCT=0. の場合と同じ扱い，ただし，指定した DMIN が欠損値で，DMAX が欠損値でない場合は CONRFCT=1. の場合と同じ扱いとなる．

コンターに付けるラベルの文字の高さ，および図の下部に入れるコンター間隔の注記の文字の高さはモジュール GGPGET/GGPSET (6.2.1) の管理するパラメータ SCONL (既定値は 0.01) およびパラメータ SREMK (既定値は 0.008) に従う．

UDPGET/UDPSET の管理する UDPACK のパラメータは，ICYCLE，RSIZEL，RSIZET を除いて有効であり，GGCNTR を呼ぶ前に UDPSET で指定できる．

6.1.8　GGMAPS [S] 地図描画

```
      SUBROUTINE GGMAPS
     I           ( MFILES )
      CHARACTER  MFILES * (NFILN)        !" 地図情報ファイル名
```

電脳ライブラリ UMPACK を用いて地図情報を出力する． これに先だって，GGAXES によりしかるべく地図座標変換が設定されている必要がある．

ファイル名 MFILES には ’,’ を区切り子として複数の地図情報ファイルを指定できる． 出力可能な地図ファイルについては電脳ライブラリのマニュアルを参照のこと．

6.1.9　GGTONE [S] トーン描画

```
      SUBROUTINE GGTONE
     I         ( HHEAD, GDATA,
     I           TONL , ITON , NTON )
*
      CHARACTER  HHEAD ( * )*(*)         ! ヘッダー
      REAL       GDATA ( * )             ! データ
      REAL       TONL  ( NTON+1 )        ! トーン境界値
      INTEGER    ITON  ( NTON )          ! トーン番号
*     INTEGER    NTON                    ! トーンの数
```

UEPACK の UETONE を用いてハーフトーンのぬりわけを行なう． これに先だってモジュール GGAXES (6.1.4) が呼ばれている必要がある．

ITON(I) は，値が TONL(I) と TONL(I+1) の間である領域にぬる濃淡パターンの番号を示す．境界のデータレベルは小さい順 (TONL(I)<TONL(I+1)) に指定しなければならない． また，TONL(1) に欠損値を入れておくと，TONL(2) 以下のデータの領域を全て ITON(1) のパターンでぬる． 同様に TONL(NTON+1) に欠損値を入れておくと，TONL(NTON) 以上のデータの領域を全て ITON(NTON) のパターンでぬる．

モジュール GGPGET/GGPSET (6.2.1) の管理するパラメータ LTONS (既定値は .TRUE.) が .TRUE. の場合には，パラメータ TXSLOC，TXYLOC の位置に，パラメータ TXSWDH，TXYWDH の幅でトーンのスケールを表示する． スケールにつく文字の大きさはパラメータ SREMK (既定値は 0.008) に従う．

6.1.10　GGCTON [S] トーン段彩

```
      SUBROUTINE GGCTON
```

```
      I         ( HHEAD , GDATA  )
*
       CHARACTER  HHEAD ( * )*(*)         !" ヘッダー
       REAL       GDATA ( * )             !" データ
```

カラーで 2 次元等値線図を色塗りする． 内部で GGSTON を呼んでトーン設定を行い，GGTONE を呼んで塗り分ける．

6.1.11 GGSTON [S] トーン段彩設定

```
       SUBROUTINE GGSTON
      I         ( HHEAD , GDATA  )
*
       CHARACTER  HHEAD ( * )*(*)         !" ヘッダー
       REAL       GDATA ( * )             !" データ
```

6.1.12 GGTONS [S] トーンスケール

```
       SUBROUTINE GGTONS
      I         ( HHEAD )
*
       CHARACTER  HHEAD ( * )*(*)         !" ヘッダー
```

6.1.13 GGVECT [S] ベクトル描画

```
       SUBROUTINE GGVECT
      I         ( HHEADU, GDATAU,
      I           HHEADV, GDATAV )
*
       CHARACTER  HHEADU ( * )*(*)        ! ヘッダー (u 成分)
       CHARACTER  HHEADV ( * )*(*)        ! ヘッダー (v 成分)
       REAL       GDATAU ( * )            ! データ (u 成分)
       REAL       GDATAV ( * )            ! データ (v 成分)
```

UPACK の UGVECT を用いてベクトル図を描画する． これに先だってモジュール GGAXES (6.1.4) が呼ばれている必要がある．

ベクトルの大きさのスケーリングは，ヘッダーの DMIN，DMAX と，モジュール GGPGET/GGPSET (6.2.1) の管理するパラメータ VECFCT (既定値は 1.0) を参照して決められる． ヘッダーの DMIN，DMAX のうち絶対値の大きな方が図上での格子点間隔の VECFCT 倍に当るように設定される． モジュール GGPGET/GGPSET (6.2.1) の管理するパラメータ LEQRAT (既定値は .FALSE.) が .TRUE. の場合は，X 方向と Y 方向のスケーリングファクターを同じにする．

DMIN，DMAX が欠損値の場合は，UGVECT の内部で適当に決める．

モジュール GGPGET/GGPSET (6.2.1) の管理するパラメータ VXINT (既定値は 1)，パラメータ VYINT (既定値は 1) が 1 以外のときには，それぞれベクトルを描く格子点を X 方向に VXINT 格子おき，Y 方向に VYINT 格子おきとして間引いて描く．

モジュール GGPGET/GGPSET (6.2.1) の管理するパラメータ LUNIT (既定値は .TRUE.) が .TRUE. の場合にはユニットベクトルを描く． その位置はモジュール GGPGET/GGPSET (6.2.1) の管理するパラメータ VXULOC (既定値は 0.12) およびパラメータ VYULOC (既定値は 0.12) で指定する．図の下部に入れるユニットベクトルの単位の注記の文字の高さはモジュール GGPGET/GGPSET (6.2.1) の管理するパラメータ SREMK (既定値は 0.008) に従う．

UGPGET/UGPSET の管理する UGPACK のパラメータは，LNRMAL，XFACT1，XFACT2，LUNIT を除いて有効であり，GGVECT を呼ぶ前に UGPSET で指定できる．

6.1.14　GGSCAT [S] 散布図描画

```
      SUBROUTINE GGSCAT
     I         ( HHEADX, GDATAX,
     I           HHEADY, GDATAY, HPOS   )
*
      CHARACTER  HHEADX( * )*(*)         !" ヘッダー
      REAL       GDATAX( * )             !" データ
      CHARACTER  HHEADY( * )*(*)         !" ヘッダー
      REAL       GDATAY( * )             !" データ
      CHARACTER  HPOS       *(*)         !" データ軸 ('X' or 'Y')
```

6.1.15　GGCURV [S] 折れ線描画

```
      SUBROUTINE GGCURV
     I         ( HHEAD , GDATA , HPOS  )
*
      CHARACTER  HHEAD ( * )*(*)         ! ヘッダー
      REAL       GDATA ( * )             ! データ
      CHARACTER  HPOS       *(*)         ! データ軸 ('X' or 'Y')
```

SGPACK のポリラインプリミティブ SGPLU を用いて折れ線図を描画する． これに先だってモジュール GGAXIS (6.1.5) が呼ばれている必要がある．

HPOS='X' の時は横軸をデータ軸として，HPOS='Y' の時は縦軸をデータ軸としてプロットする．

USPACK を利用してデータ軸のウインドウを設定し，座標軸を描く． 軸の範囲はヘッダーの記述子 ASTR1，ASTR2 に準拠し，さらに，ヘッダーの記述子 DMIN，DMAX，DIVS，DIVL を参考にする． これらに欠損値以外が設定されている場合には，DMIN から DMAX までのウィンドウを設定し，DIVS ごとに目盛を打ち，DIVL ごとにラベルを付ける． それ以外のときは USPACK によって適当に決める．

軸に付けるラベルの文字の高さ，および軸に付けるタイトルの文字の高さはモジュール GGPGET/GGPSET (6.2.1) の管理するパラメータ SAXSL (既定値は 0.015) およびパラメータ SAXST (既定値は 0.015) に従う．

軸の向き，および通常の軸か対数軸かは記述子 STYP(1.2.2 参照) に従う

折れ線の属性，すなわちラインタイプ (実線か点線かなど)，ラインインデックス (太さあるいは色) は，それぞれ SGPACK のモジュール SGSPLT，SGSPLI で，GGCURV を呼ぶ前に設定する．

複数の折れ線を 1 枚の図に描く場合には連続して GGCURV を呼べばよい． 2 本目以降はウインドウは変化しない． 全ての折れ線が図に入るようにウィンドウをスケーリングするためには，あらかじめ全てのデータに対してモジュール GGRANG (6.1.18) を呼んでおく必要がある．

6.1.16　GGMARK [S] マーク描画

```
      SUBROUTINE GGMARK
     I         ( HHEAD , GDATA , HPOS  )
*
      CHARACTER  HHEAD ( * )*(*)         ! ヘッダー
      REAL       GDATA ( * )             ! データ
      CHARACTER  HPOS       *(*)         ! データ軸 ('X' or 'Y')
```

SGPACK のポリマーカープリミティブ SGPMU を用いてマーカー図を描画する． これに先だってモジュール GGAXIS (6.1.5) が呼ばれている必要がある．

HPOS='X' の時は横軸をデータ軸として，HPOS='Y' の時は縦軸をデータ軸としてプロットする．

USPACK を利用してデータ軸のウインドウを設定し，座標軸を描く． 軸の範囲はヘッダーの記述子 ASTR1，ASTR2 に準拠し，さらに，ヘッダーの記述子 DMIN，DMAX，DIVS，DIVL を参考にする． これらに欠損値以外が設定されている場合には，DIVS ごとに目盛を打ち，DIVL ごとにラベルを付ける． それ以外のときは USPACK によって適当に決める．

軸に付けるラベルの文字の高さ，および軸に付けるタイトルの文字の高さはモジュール GGPGET/GGPSET (6.2.1) の管理するパラメータ SAXSL (既定値は 0.015) およびパラメータ SAXST (既定値は 0.015) に従う．

軸の向き，および通常の軸か対数軸かは記述子 STYP(1.2.2 参照) に従う

マーカーの属性，すなわちマーカータイプ (マークの種類)，マーカーインデックス (太さあるいは色)，マーカーの大きさは，それぞれ SGPACK のモジュール SGSPMT，SGSPMI，SGSPMS で，GGMARK を呼ぶ前に設定する．

複数のデータのマーカーを 1 枚の図に描く場合には連続して GGMARK を呼べばよい． 2 本目以降はウインドウは変化しない． 全てのマーカーが図に入るようにウィンドウをスケーリングするためには，あらかじめ全てのデータに対してモジュール GGRANG (6.1.18) を呼んでおく必要がある．

6.1.17　GGLINT [S] 線種タイトル表示

```
      SUBROUTINE GGLINT
     I         ( HHEAD , HTYPE )
*
      CHARACTER  HHEAD ( * )*(*)         !" ヘッダー
      CHARACTER  HTYPE      *(*)         !" 種類 (CURV/MARK)
```

6.1.18　GGRANG [ES] 軸の座標範囲の設定

```
      ENTRY      GGRANG
     I         ( HHEAD , GDATA , HPOS  )
*
      CHARACTER  HHEAD ( * )*(*)         ! ヘッダー
      REAL       GDATA ( * )             ! データ
      CHARACTER  HPOS *(*)               ! セットする軸 ('X' or 'Y')
```

折れ線あるいはマーカー図を 1 つの枠に複数重ねて描く場合に，全ての折れ線 (またはマーカー) が図の中に入るようにあらかじめデータ軸のスケーリングの範囲 (データの最大値，最小値) を設定するのに用いる． モジュール GGLAY1 (6.1.3) を呼んだ後，かつモジュール GGCURV (6.1.15) あるいはモジュール GGMARK (6.1.16) を呼ぶ前に呼ぶこと． 前の呼び出しの結果の最大値，最小値を含めて最大，最小を求めて値を設定するので，何度も続けて呼べば一連のデータの中の最大値，最小値が記憶される． モジュール GGAXRS (6.3.4)(GGLAY1 の中で呼ばれる) でこの設定はキャンセルされる．

例えば 2 本の折れ線を描くには次のようにする．

```
CALL GGLAY1 ( HHEAD1 )
CALL GGAXIS ( HHEAD1, 'X' )
CALL GGRANG ( HHEAD1, GDATA1, 'Y' )
CALL GGRANG ( HHEAD1, GDATA2, 'Y' )
CALL SGSPLI ( 1 )                    ! ラインインデックス 1
CALL GGCURV ( HHEAD1, GDATA1, 'Y' )
CALL SGSPLI ( 2 )                    ! ラインインデックス 2
CALL GGCURV ( HHEAD2, GDATA2, 'Y' )
```

6.2　描画パラメーター管理モジュール

次のような描画に関する内部パラメータを管理している.

パラメーター名	種類	既定値	関係モジュール	説明
レイアウト				
VRATIO	R	0.70	GGLAY1	全体の縦横比
VXMIN	R	0.20	GGLAY1	ビューポート Xmin
VXMAX	R	0.80	GGLAY1	ビューポート Xmax
VYMIN	R	0.20	GGLAY1	ビューポート Ymin
VYMAX	R	0.50	GGLAY1	ビューポート Xmax
STITL	R	0.02	GGLAY1	タイトルの文字高
SEDIT	R	0.013	GGLAY1	編集略記述の文字高
SDESC	R	0.01	GGLAY1	そのほかの記述の文字高
SAXSL	R	0.015	GGAXES 等 (*)	軸ラベルの文字高
SAXST	R	0.015	GGAXES 等 (*)	軸タイトル文字高
SREMK	R	0.008	GGCNTR 等 (**)	下部注記の文字高
等値線				
SCONL	R	0.01	GGCNTR	コンターラベルの文字高
ICYCLE	I	4	GGCNTR	コンターラベルをつける間隔
NCONMAX	I	50	GGCNTR	コンターの最大本数
CONRFCT	R	0.5	GGCNTR	コンターが多すぎる場合の省略用ファクター
ベクトル場				
VXINT	I	1	GGVECT	ベクトルの間引き間隔 (X 方向)
VYINT	I	1	GGVECT	ベクトルの間引き間隔 (Y 方向)
VECFCT	R	1.0	GGVECT	ベクトルのスケーリングのファクター
LEQRAT	L	.FALSE.	GGVECT	ベクトルのスケーリングを縦横同じにするか否か
LUNIT	L	.TRUE.	GGVECT	ユニットベクトルを描くか否か
VXULOC	R	0.10	GGVECT	ユニットベクトルの位置 (X)
VYULOC	R	0.20	GGVECT	ユニットベクトルの位置 (Y)
トーン				
LTONS	L	.TRUE.	GGTONE	トーンスケール描画スイッチ
TXSLOC	R	0.05	GGTONE	トーンスケールの位置 (X)
TYSLOC	R	0.1	GGTONE	トーンスケールの位置 (Y)
TXSWDH	R	0.25	GGTONE	トーンスケールの幅 (X)
TYSWDH	R	0.05	GGTONE	トーンスケールの幅 (Y)
TLNUM	I	4	GGTONS	トーンラベルの数
LLINT	L	.TRUE.	GGCURV 等 (***)	線種コメント
LTVX1	R	0.22	GGLINT	〃 位置：x (1)
LTVX2	R	0.55	GGLINT	〃 位置：x (2)
LTWDH	R	0.08	GGLINT	〃 幅
INNER	I	-1	GGAXSZ	目盛位置
CONTSET	I	0	GGCNTR	コンターレベルセット
TONESET	I	0	GGTONE 等	トーンレベルセット
NTONE	I	50	GGTONE 等	トーンレベル数
TONEPAT	I	11999	GGSTON	トーンレベル開始
TONEINC	I	1000	GGSTON	トーンレベル間隔
地図投影				
MAPPRJ	I	0	GGAXES	投影法
MAPLON	R	180	GGAXES	投影中心経度
MAPLAT	R	0.0	GGAXES	投影中心緯度
MAPROT	R	0.0	GGAXES	投影回転角度
MAPFAC	R	1.0	GGAXES	投影拡大率

(*) : GGAXES, GGAXIS, GGCURV, GGMARK　　(**): GGCNTR, GGVECT, GGTONE (***): GGCURV, GGMARK, GGLAY3

数字/論理パラメータに関してはモジュール GGPGET/GGPSET (6.2.1) で, 文字パラメータに関してはモジュール GGCGET/GGCSET (6.2.3) で, それぞれ参照/設定できる.

6.2.1 GGPGET [S] 描画パラメータ（数）を参照

```
      SUBROUTINE GGPGET
     I         ( HP,
     O           IPARA )
*
      CHARACTER HP    *(*)                ! パラメーターの名前
*     (INTEGER)  IPARA                    ! パラメーターの内容：数字
```

パラメーター HP の内容を IPARA に入れる． IPARA の実引数としては，適当な型の変数を用いること (1.7.2 参照).

6.2.2 GGPSET [ES] 描画パラメータ（数）を設定

```
      ENTRY GGPSET
     I         ( HP, IPARA )
*
      CHARACTER HP    *(*)                ! パラメーターの名前
*     (INTEGER)  IPARA                    ! パラメーターの内容：数字
```

IPARA をパラメーター HP に入れる． IPARA の実引数としては，適当な型の変数/定数を用いること (1.7.2 参照).

6.2.3 GGCGET [ES] 描画パラメータ（文字）を参照

```
      ENTRY GGCGET
     I         ( HP,
     O           HPARA )
*
      CHARACTER HP    *(*)                ! パラメーターの名前
      CHARACTER HPARA*(*)                 ! パラメーターの内容：文字
```

パラメーター HP の内容を HPARA に入れる.

6.2.4 GGCSET [ES] 描画パラメーター（文字）を設定

```
      ENTRY GGCSET
     I         ( HP, HPARA )
*
```

```
      CHARACTER HP    *(*)                ! パラメーターの名前
      CHARACTER HPARA*(*)                 ! パラメーターの内容：文字
```

HPARA をパラメーター HP に入れる.

6.3　描画下請けモジュール

以下のモジュールは, 基本描画モジュールから呼ばれるもので通常ユーザーが直接呼ぶ必要はない.

6.3.1　GGPDSC [S] ヘッダーの内容 (実数) を文字化して描く

```
      SUBROUTINE GGPDSC
     I         ( HHEAD, HP   , VX   , VY    )
*
      CHARACTER  HHEAD ( * )*(*)             ! ヘッダー
      CHARACTER  HP         *(*)             ! 記述子の名称
      REAL       VX                          ! X 座標 (NDC)
      REAL       VY                          ! Y 座標 (NDC)
```

位置 (VX,VY) に, ヘッダーの実数の記述子 HP の内容を文字化して描く.　文字化の形式は UTIL1 のモジュール CHVAL の'D' フォーマットに従う.

6.3.2　GGCDSC [ES] ヘッダーの内容 (文字) を描く

```
      ENTRY      GGCDSC
     I         ( HHEAD, HP   , VX   , VY    )
*
      CHARACTER  HHEAD ( * )*(*)             ! ヘッダー
      CHARACTER  HP         *(*)             ! 記述子の名称
      REAL       VX                          ! X 座標 (NDC)
      REAL       VY                          ! Y 座標 (NDC)
```

位置 (VX,VY) に, ヘッダーの文字の記述子 HP の内容を描く.

6.3.3　GGAXSZ [S] 1 つの軸を描き座標設定

```
      SUBROUTINE GGAXSZ
     I         ( HHEAD , AXISZ , HPOS   )
*
      CHARACTER  HHEAD ( * )*(*)          ! ヘッダー
      REAL       AXISZ ( * )              ! 格子位置
      CHARACTER  HPOS *(*)                ! セットする軸 ('X' or 'Y')
```

座標を設定し，座標軸を描く． これに先だってモジュール GGLAY1 (6.1.3) が呼ばれている必要がある．

HPOS='X' の時は横軸のみ，HPOS='Y' の時は縦軸のみ設定する．

USPACK を利用してウインドウを設定し，座標軸を描く． 軸の範囲はヘッダーの記述子 ASTR1，ASTR2 に準拠し，さらに，ヘッダーの記述子 DMIN，DMAX，DIVS，DIVL を参考にする． これらに欠損値以外が設定されている場合には，DMIN から DMAX までのウィンドウを設定し，DIVS ごとに目盛を打ち，DIVL ごとにラベルを付ける． それ以外のときは USPACK によって適当に決める．

軸に付けるラベルの文字の高さ，および軸に付けるタイトルの文字の高さはモジュール GGPGET/GGPSET (6.2.1) の管理するパラメータ SAXSL (既定値は 0.015) およびパラメータ SAXST (既定値は 0.015) に従う．

軸の向き，および通常の軸か対数軸かは記述子 STYP(1.2.2 参照) に従う

6.3.4　GGAXRS [ES] 軸描画のリセット

```
      ENTRY      GGAXRS
```

軸描画モジュール GGAXSZ (6.3.3) の内部パラメータをリセットする． モジュール GGLAY1 (6.1.3) で呼ばれているのでユーザーが通常使用することはないが，モジュール GGLAY1 (6.1.3) に相当するモジュールをユーザーが作成する場合は，ウィンンドウを設定し直す度に入れること．

6.3.5　GGAXLZ [S] 軸ラベルの描画フラグ

```
      SUBROUTINE GGAXLZ ( HPOS )
*
      CHARACTER  HPOS   * 1
```

6.3.6　GGAXIX [S] 軸のセット X or Y 軸のみ : 拡張

```
      SUBROUTINE GGAXIX
     I         ( HHEAD , HPOS  ,
     I           DMIN  , DMAX  , DIVS  , DIVL  )
*
      CHARACTER  HHEAD ( * )*(*)         !" ヘッダー
      CHARACTER  HPOS *(*)               !" セットする軸 ('X' or 'Y')
```

6.3.7　PORAXZ [S] 外枠囲み (極座標)

```
      SUBROUTINE PORAXZ                  !" 外枠囲み (極座標)
     I             ( HHEADX , AXISX ,
     I               HHEADY , AXISY   )
*
```

```
      CHARACTER  HHEADX( NDC )*(NCC)     !" 動径軸ヘッダーファイル
      CHARACTER  HHEADY( NDC )*(NCC)     !" 方位角軸ヘッダーファイル
*
      REAL       AXISX(*)                !" X 座標値
      REAL       AXISY(*)                !" Y 座標値
```

6.3.8　GGLTRS [ES] 線種タイトル表示リセット

```
      ENTRY      GGLTRS
```

第7章　ユーティリティーモジュール群

下請け処理を行なうモジュール群である.

7.1　データのサイズの参照等のモジュール

ヘッダーからデータのサイズを参照するモジュールが用意されている.

7.1.1　GUQSIZ [S] データサイズの取得

```
      SUBROUTINE GUQSIZ
     I         ( HHEAD ,
     O           IXSTR , IXEND , IXDIM ,
     O           IYSTR , IYEND , IYDIM ,
     O           IZSTR , IZEND , IZDIM  )
*
      CHARACTER  HHEAD ( * )*(*)         ! ヘッダー
      INTEGER    IXSTR , IXEND , IXDIM   ! 軸１の始め 終り 寸法
      INTEGER    IYSTR , IYEND , IYDIM   ! 軸２の始め 終り 寸法
      INTEGER    IZSTR , IZEND , IZDIM   ! 軸３の始め 終り 寸法
```

ヘッダーから，データのサイズを読み取る． ここで IXSTR，IXEND 等は，それぞれヘッダーの記述子 ASTR1，AEND1 等の内容であり，IXDIM = IXEND-IXSTR+1 である.

格子番号（格子位置データとの対応を示す）IXSTR から IXEND までのデータが入っていることを示す.

7.1.2　GUCSIZ [S] データサイズの比較

```
      SUBROUTINE GUCSIZ
     I         ( HHEAD1, HHEAD2,
     O           IXSTR1, IXDIM1, IXSTR2, IXDIM2,
     O           IYSTR1, IYDIM1, IYSTR2, IYDIM2,
     O           IZSTR1, IZDIM1, IZSTR2, IZDIM2,
     O           IXSTRC, IXENDC, IXDIMC,
     O           IYSTRC, IYENDC, IYDIMC,
     O           IZSTRC, IZENDC, IZDIMC         )
*
      CHARACTER  HHEAD1 ( * )*(*)        ! ヘッダー１
      CHARACTER  HHEAD2 ( * )*(*)        ! ヘッダー２
```

```
      INTEGER    IXSTR1, IXDIM1          ! 軸１の始めと大きさ
      INTEGER    IXSTR2, IXDIM2          !  〃  （ヘッダー２）
      INTEGER    IYSTR1, IYDIM1          ! 軸２の始めと大きさ
      INTEGER    IYSTR2, IYDIM2          !  〃  （ヘッダー２）
      INTEGER    IZSTR1, IZDIM1          ! 軸３の始めと大きさ
      INTEGER    IZSTR2, IZDIM2          !  〃  （ヘッダー２）
      INTEGER    IXSTRC, IXENDC, IXDIMC  ! 軸１の共通部分
      INTEGER    IYSTRC, IYENDC, IYDIMC  ! 軸２の共通部分
      INTEGER    IZSTRC, IZENDC, IZDIMC  ! 軸３の共通部分
```

ヘッダーから，2 つのデータのサイズを読み取り，共通部分を抽出する． ここで IXSTR1, IXDIM1 は，それぞれヘッダー 1 の記述子 ASTR1 等の内容，および ((AEND1 の内容)-(ASTR1 の内容)+1) である． 格子番号 (格子位置データとの対応を示す) IXSTR から IXDIM 個のデータが入っていることを示す． IXSTR2，IXDIM2 は同様にヘッダー 2 に対応するものである．

IXSTRC，IXENDC，IXDIMC は，ヘッダー 1 とヘッダー 2 から取得されるデータ領域の重なりの範囲を示す． 格子番号 IXSTRC から IXENDC まで IXDIMC 個のデータは，ヘッダー 1 とヘッダー 2 に対応するデータに共通して存在することを示す．

7.1.3　GUAXCK [S] 格子名称のチェック

```
      SUBROUTINE GUAXCK
     I         ( HHEAD1, HHEAD2,
     O           ICHK            )
*
      CHARACTER  HHEAD1 ( * )*(*)        ! ヘッダー 1
      CHARACTER  HHEAD2 ( * )*(*)        ! ヘッダー 1
      INTEGER    ICHK                    ! 格子名が合致していれば 0
```

ヘッダー 1 とヘッダー 2 の 3 つの軸の格子識別名称が対応しているかどうか調べる． 全て合致している場合のみ ICHK として 0 を返す． そうでない場合は，軸 1 が違っている場合は 1 を，軸 2 が違っている場合は 2 を，軸 3 が違っている場合は 4 を，2 つ以上の軸が違っていればその和を返す．

7.2　格子情報に関するモジュール

格子情報を扱うモジュールが用意されている.

7.2.1　GUQAXV [S] 格子情報の取得

```
      SUBROUTINE GUQAXV
     I         ( HHEAD , ID    , HKIND ,
     O           HHEADA, AXIS  , IEOD  )
*
      CHARACTER  HHEAD ( * )*(*)             ! ヘッダー
```

```
      INTEGER    ID                          ! 何次元目の軸か
      CHARACTER  HKIND      *(*)             ! LOC/WGT
      CHARACTER  HHEADA( * )*(*)             ! ヘッダー
      REAL       AXIS   ( * )                ! 位置/重み
      INTEGER    IEOD                        ! データなしフラグ
```

ヘッダー HHEAD の ID 次元めの軸に対応する格子情報を取得する．

HKIND が'LOC' の場合は格子位置を，HKIND が'WGT' の場合は格子重みを取得する．

取得される格子位置/重みデータの範囲は，ヘッダー HHEAD の ASTRx，AENDx（x は ID の内容) に従う．

該当する軸の格子識別名称が文字'@' で始まっているときは，内部設定格子であると判断する．

該当する格子情報が存在しない場合あるいは格子識別名称として空白を指定した場合には，IEOD≠0 を返す． 正常終了の場合は IEOD=0 を返す．

内部でモジュール GUQAXS (7.2.3) を呼んでいる．

7.2.2　GUQAXD [S] 格子のサイズの参照

```
      SUBROUTINE GUQAXD
     I         ( HAXIS ,
     O           IADIM , OCYCL  )
*
      CHARACTER  HAXIS      *(*)             ! 格子識別名称
      INTEGER    IADIM                       ! 格子のサイズ
      LOGICAL    OCYCL                       ! サイクリックか？
```

格子識別名称 HAXIS の格子情報から，全格子数とサイクリックな座標軸であるかどうかを参照する．

格子識別名称が文字'@' で始まっているときは，内部設定格子であると判断する．

全格子数は格子情報ヘッダーの記述子 AEND1 の内容である． サイクリックな座標軸であるかどうかは，記述子 DSET の内容の 1 文字目が'C' であるかどうかによって判断する．

7.2.3　GUQAXS [S] 格子情報ファイルの読み込み

```
      SUBROUTINE GUQAXS
     I         ( HAXIS , HKIND ,
     O           HHEAD , AXIS   , IEOD  )
*
      CHARACTER  HAXIS *(*)                  ! 格子識別名称
      CHARACTER  HKIND *(*)                  ! LOC/WGT
      CHARACTER  HHEAD ( * )*(*)             ! ヘッダー
      REAL       AXIS   ( * )                ! 位置/重み
      INTEGER    IEOD                        ! データなしフラグ
```

格子識別名称 HAXIS の格子情報を取得する．

HKIND が'LOC' の場合は格子位置を，HKIND が'WGT' の場合は格子重みを取得する．

取得される格子位置/重みデータの範囲は，設定されている格子情報全体である．

格子識別名称が文字'@'で始まっているときは内部設定格子であると判断し，モジュール GUQIAX (7.3.4) を呼んでいる．

該当する格子情報が存在しない場合あるいは格子識別名称として空白を指定した場合には，IEOD≠0 を返す． 正常終了の場合は IEOD=0 を返す．

7.2.4　GUEAXN [S] 軸編集記述を作成

```
      SUBROUTINE GUEAXN
     I         ( HAXIS , HED   , HET   ,
     I           ISTR  , IEND  ,
     O           HEDIT , HETTL           )
*
      CHARACTER  HAXIS         *(*)          ! 格子識別名称
      CHARACTER  HED           *(*)          ! 編集記述子接頭
      CHARACTER  HET           *(*)          ! 編集タイトル接尾
      INTEGER    ISTR                        ! 格子番号始め
      INTEGER    IEND                        ! 格子番号終り
      CHARACTER  HEDIT         *(*)          ! 編集記述子
      CHARACTER  HETTL         *(*)          ! 編集タイトル
```

ある軸に対する操作の編集記述文字列を作成する． モジュール GPXRDC (5.1.2) 等で呼ばれる．

ISTR≠IEND の時は編集略記号として HED の内容と ISTR と IEND の内容を連接したもの (ISTR と IEND の間に'-' が入る) が，編集タイトルとして '(HAXIS の内容)=(格子番号 ISTR の位置)-(格子番号 IEND の位置)' (例えば GLON128=30.-60.) が作成される．

ISTR=IEND の時は編集略記号として HED の内容と ISTR の内容を連接したものが，編集タイトルとして '(HAXIS の内容)=(格子番号 ISTR の位置)' (例えば GLON128=30.) が作成される．

該当する格子情報が存在しない場合，あるいは格子識別名称として空白を指定した場合には，編集タイトルは HAXIS の内容に HET の内容を連接されたものになる．

内部でモジュール GUQAXV (7.2.1) が呼ばれ，格子位置情報が参照される．

7.3　内部設定格子情報に関するモジュール

内部設定格子情報を扱うモジュールが用意されている．

7.3.1　GUSIAX [ES] 内部設定格子の設定

```
      ENTRY      GUSIAX
     I         ( HHEAD , AXLOC , AXWGT ,
     I           INUM                  )
```

```
*
      CHARACTER  HHEAD ( * )*(*)          ! ヘッダー
      REAL       AXLOC ( * )              ! 位置
      REAL       AXWGT ( * )              ! 重み
      INTEGER    INUM                     ! 内部設定格子番号
```

内部設定格子情報をメモリー上に記憶させる． 格子位置情報および格子重み情報を対で同時に登録する． ヘッダーの中の格子識別名称 ITEM としては，文字'@' で始まるものを指定すること． 本来は格子情報のヘッダーは，格子位置情報と格子重み情報とで記述子 DSET が違うはずであるが，入力する HHEAD はどちらに対応するものでもよい．

内部設定格子は複数登録できる． 標準では 10 個までである． INUM で，いくつめの格子情報として登録するかを指定する．

7.3.2　GUAIAX [ES] 内部設定格子位置の付加設定

```
      ENTRY      GUAIAX
     I         ( HAXIS , AXLOC1, AXWGT1,
     I           IZ                     )
*
      CHARACTER  HAXIS      *(*)         ! 格子識別名称
      REAL       AXLOC1                  ! 位置
      REAL       AXWGT1                  ! 重み
      INTEGER    IZ                      ! 格子番号
```

格子識別名称 HAXIS の内部設定格子情報の 1 つの格子の位置と重みを変更する． 格子位置情報および格子重み情報を対で同時に指定する． HAXIS としては，文字'@' で始まるものを指定すること． これに先だって，同じ格子識別名称でモジュール GUSIAX (7.3.1) が呼ばれている必要がある．

IZ として，現在設定されているサイズよりも大きな値を指定した場合には，サイズが IZ に変更される． また，IZ=0 を指定した場合には，IZ に (現在設定されているサイズ+1) が指定されたものとして扱う．

7.3.3　GUAIAH [ES] 内部設定格子位置の付加設定 (2)

```
      SUBROUTINE GUAIAH
     I         ( HHEAD , ID    ,
     I           AXLOC1, AXWGT1,
     I           IZ                     )
*
      CHARACTER  HHEAD ( * )*(*)          ! ヘッダー
      INTEGER    ID                       ! 何次元目の軸か
      REAL       AXLOC1                   ! 位置
      REAL       AXWGT1                   ! 重み
```

```
      INTEGER    IZ                        ! 格子番号
```

ヘッダー HHEAD の ID 次元めの軸に対応する内部設定格子情報の 1 つの格子の位置と重みを変更する． 格子位置情報および格子重み情報を対で同時に指定する． これに先だって，同じ格子識別名称でモジュール GUSIAX (7.3.1) が呼ばれている必要がある．

そのほかはモジュール GUAIAX (7.3.2) と同様．

7.3.4　GUQIAX [S] 内部設定格子の参照

```
      SUBROUTINE GUQIAX
     I         ( HAXIS , HKIND ,
     O           HHEAD , AXIS  , IEOD   )
*
      CHARACTER  HAXIS      *(*)         ! 格子識別名称
      CHARACTER  HKIND *(*)              ! LOC/WGT
      CHARACTER  HHEAD ( * )*(*)         ! ヘッダー
      REAL       AXIS  ( * )             ! 位置/重み
      INTEGER    IEOD                    ! データなしフラグ
```

格子識別名称 HAXIS の内部設定格子情報を取得する．

HKIND が'LOC' の場合は格子位置を，HKIND が'WGT' の場合は格子重みを取得する．

取得される格子位置/重みデータの範囲は，設定されている格子情報全体である．

該当する格子情報が存在しない場合あるいは格子識別名称として空白を指定した場合には，IEOD≠0 を返す． 正常終了の場合は IEOD=0 を返す．

このモジュールは，格子識別名称 HAXIS が文字'@' で始まっているときにモジュール GUQAXS (7.2.3) から呼ばれる． ユーザーがこのモジュールを直接呼ぶ必要はない．

7.3.5　GUQIAH [ES] 内部設定格子ヘッダーの参照

```
      ENTRY      GUQIAH
     I         ( HAXIS ,
     O           HHEAD , IEOD   )
*
      CHARACTER  HAXIS      *(*)         ! 格子識別名称
      CHARACTER  HHEAD ( * )*(*)         ! ヘッダー
      INTEGER    IEOD                    ! データなしフラグ
```

格子識別名称 HAXIS の内部設定格子情報のヘッダー部分を取得する．

該当する格子情報が存在しない場合あるいは格子識別名称として空白を指定した場合には，IEOD≠0 を返す． 正常終了の場合は IEOD=0 を返す．

7.4　配列処理のためのユーティリティーモジュール

7.4.1　GUSZCK [S] 配列の大きさのチェック

```
      SUBROUTINE GUSZCK
     I         ( HHEAD , ISIZEO  )
*
      CHARACTER  HHEAD ( * )*(*)             ! ヘッダー
      INTEGER    ISIZEO                      ! 出力の大きさ
```

配列の大きさをチェックする． HHEAD の中の記述子 SIZE に設定されているサイズが ISIZEO よりも小さい場合にエラーを表示して停止する． ただし，モジュール GTPGET/GTPSET (2.1.1) の管理するパラメータ SUBCHK (既定値は .FALSE.) が .TRUE. の場合にのみ有効である． 1.5 を参照のこと．

7.4.2　GUSZCZ [S] 大きさのチェック

```
      SUBROUTINE GUSZCZ
     I         ( ISIZEA, ISIZEO )
*
      INTEGER    ISIZEA                      ! 領域の大きさ
      INTEGER    ISIZEO                      ! 出力の大きさ
```

配列の大きさをチェックする． ISIZEA が ISIZEO よりも小さい場合にエラーを表示して停止する．

7.4.3　GUSMIS [S] 欠損値を指定

```
      SUBROUTINE GUSMIS
     I         ( HHEAD  )
*
      CHARACTER  HHEAD  ( * )*(*)        ! ヘッダー
```

HHEAD の記述子 MISV に書かれている欠損値の値を GLPGET/GLPSET の管理する SUBFUNC のパラメータ RMISS に設定し，あわせて LMISS を .TRUE. に設定して欠損値処理を有効にする．

7.5　時刻に関するユーティリティーモジュール

7.5.1　GUQTIM [S] 時刻の参照

```
      SUBROUTINE GUQTIM
     I         ( HHEAD ,
     O           RT    , HUTIM )
*
```

```
      CHARACTER  HHEAD ( * )*(*)          ! ヘッダー
      REAL       RT                       ! 時刻
      CHARACTER  HUTIM      *(*)          ! 時刻単位
```

HHEAD の記述子 TIME に書かれている時刻をモジュール GTCGET/GTCSET (2.1.3) の管理するパラメータ UTIM (既定値は 'DAY') の単位の実数に直して RT にセットする． HUTIM にはパラメータ UTIM の内容が入る．

7.5.2　GUCTIM [S] 時間単位の変換 HUNIT → HUNITY

```
      SUBROUTINE GUCTIM
     I         ( KT    , HUNIT  , HUNITY, IDELT  ,
     O           RT                               )
*
      INTEGER    KT               ! 入力時刻：単位 HUNIT
      CHARACTER  HUNIT *(*)       ! 入力 KT の単位
      CHARACTER  HUNITY*(*)       ! 変換すべき単位
      INTEGER    IDELT            ! １ステップの秒数
      REAL       RT               ! 出力時刻：単位 HUNITY
```

単位 HUNIT の整数で表された時刻 KT を HUNITY で指定する単位の実数に直して RT にセットする．

HUNITY の内容が'STEP' の場合には，KT を秒に変換したものを IDELT で割ったものを RT とする．

7.5.3　GUQNOW [S] 現在の日付・時刻 yyyymmddhhmmss

```
      SUBROUTINE GUQNOW
     O         ( HTIME )
*
      CHARACTER  HTIME  *(*)            ! 時刻 (yyyymmddhhmmss)
```

現在の日付・時刻を (yyyymmddhhmmss) の形で参照する．

7.5.4　GUSNOW [S] 現在の時刻を作成時刻に設定

```
      SUBROUTINE GUSNOW
     M         ( HHEAD ,
     I           HP     )
*
      CHARACTER  HHEAD  ( * )*(*)       ! ヘッダー
      CHARACTER  HP          *(*)       ! 記述子名
```

現在の日付，時刻を HHEAD の記述子 HP に設定する．

7.5.5　GUTPAC [S] 時刻をパック

```
      SUBROUTINE GUTPAC
     I         ( IYEAR , IMONTH, IDAY  ,
     I           IHOUR , IMIN  , ISEC  ,
     O           HDATE                 )
*
      INTEGER    IYEAR                        ! 年
      INTEGER    IMONTH                       ! 月
      INTEGER    IDAY                         ! 日
      INTEGER    IHOUR                        ! 時
      INTEGER    IMIN                         ! 分
      INTEGER    ISEC                         ! 秒
      CHARACTER  HDATE *(*)                   ! 時刻 (yyyymmddhhmmss)
```

IYEAR～ISEC で指定される日付・時刻を (yyyymmddhhmmss) の形にして HDATE に入れる.

7.5.6　GUTUPC [ES] 時刻をアンパック

```
      ENTRY      GUTUPC
     O         ( IYEAR , IMONTH, IDAY  ,
     O           IHOUR , IMIN  , ISEC  ,
     I           HDATE                 )
*
      INTEGER    IYEAR                        ! 年
      INTEGER    IMONTH                       ! 月
      INTEGER    IDAY                         ! 日
      INTEGER    IHOUR                        ! 時
      INTEGER    IMIN                         ! 分
      INTEGER    ISEC                         ! 秒
      CHARACTER  HDATE *(*)                   ! 時刻 (yyyymmddhhmmss)
```

(yyyymmddhhmmss) の形で表された日付・時刻 HDATE を IYEAR～ISEC にばらして取り出す.

7.6　その他のユーティリティーモジュール

7.6.1　GUCTXT [S] エスケープシーケンス正規化

```
      SUBROUTINE GUCTXT
     I         ( HEXT  ,
     O           HNORM  )
*
      CHARACTER HEXT  *(*)          ! エスケープ入り文字列
      CHARACTER HNORM *(*)          ! 正規化された文字列
```

HEXT の中の'\'+3 桁の数字を文字 CSGI(n)(n は 3 桁の数字が 10 進数として表す数) に置き換えたものを HNORM に入れる． \ 以外の文字はそのままコピーされる． \ 自身を表すときは '\ \' のように 2 つ続けて入れておく．

例えば，'\170=\ \100． \ 016' は，CSGI(170)//'=\ 100． '//CSGI(16) となる．

7.7　既存パッケージへの追加

GTOOL3 の機能実現のために, MATH1 レベルに相当する次のようなモジュールが追加されている.

7.7.1　GUVINT/GUVDIN/GUVOUT [S] ベクトルデータの重みつき平均

```
      SUBROUTINE GUVINT(WZ,WW,IX)
      ENTRY      GUVDIN(WZ,WW,IX,X,WT)
      ENTRY      GUVOUT(WZ,WW,IX)
*
      REAL      WZ(IX)  ! 出力用配列
      REAL      X (IX)  ! 入力データ
      REAL      WW(IX)  ! 重みを足したものを入れておく配列
      REAL      WT      ! 重み
```

1 種類のベクトルデータを，重みを指定して連続的に読み込んで荷重平均を求める． GUVINT は初期化を行なう． GUVDIN はデータを読み込む． GUVOUT は結果を求める．

第8章　サンプルプログラム

8.1　一般的使用法

それぞれ

```
% gtxxxx [filename...] [options...]
```

のように実行する.
各オプションは, 次の様に指定する.

文字型	:	`title:Temperature` または `"title:Averaged Temp."`
文字型以外	:	`z=1`
論理型	:	`-mono` (`mono=T` と同じ), `#clabel` (`clabel=F` と同じ)

オプション指定は省略できる.
また, 複数の引数をもつ配列型オプションでは, cont=10,50 のように指定する (カンマの前後に空白をいれないこと). 2 つめ以降は省略可である. cont(2)=50 あるいは cont=,50 のような指定も可.
=, : を含まず, - もしくは # で始まらないトークン文字列 (空白もしくは改行にはさまれた文字列) はファイル名とみなされる. ファイル名はオプション列のどこにあってもよい.
以下, x,y,z 軸とはデータの第 1,2,3 次元をさす.
環境変数 GTTMPDIR を設定することにより, 中間ファイル (gtool.out) をカレント以外のディレクトリに作ることができる.

8.2　コマンド一覧

8.2.1　表示・描画

`gtcont` 等値線図, トーン

`gtvect` ベクトル図

`gtcurv` 折れ線図

`gtmark` 折れ線図

`gtscat` 散布図

`gtshow` キャラクタ表示

8.2.2　数学

gtadd 和

gtsub 差

gtmlt 積

gtdiv 商

gtsqr 平方根

gtavr 系列平均

gtedy eddy

gtimis 欠損値補間

gtmask マスキング

gtfunc 関数化前処理

gtbpf band-pass filter

gtxpwspct x 軸パワースペクトル

gtdump 球面調和関数フィルタ

gtint1 int 1 -¿gtool3

gtint2 int 2 -¿gtool3

8.2.3　gtool 形式データ処理

gtaxis 軸ファイル抽出

gtcrax 書式つき-¿gtool3 軸ファイル

gtskelton gtcreate の skelton

gtcon スカラー値セット

gtset 値のセット

gtcreate 書式つき-¿gtool3

gthead ヘッダの変更

gtext サイズの変更, 次元の追加

gtinterp 格子変換

gtsel 選択, フィールド変更

gtseq 系列をまとめたデータの作成

gtcat gtool ファイルの結合

gthbin HITAC VBS 変換

8.2.4　大気物理

gthydro T -¿z hydrostatic

gtslp 海面更正気圧

gtstrm 質量流線関数

gtqsat Qsat

gtrelh 相対湿度

gts2p σ-¿p

gtthet 温位¡-¿温度

gtthetae 相当温位

gttv 仮温度

8.3　主なコマンドの使用法

8.3.1　gtcont : 2 次元等値線描画, トーン塗分け

```
&option
x=  -1,                     切り出す x 格子番号 0 なら，x 平均
y=  -1,                     切り出す y 格子番号 0 なら，y 平均
z=  -1,                     切り出す z 格子番号 0 なら，z 平均
cont=   -999.000   -999.000, コンター間隔，コンターラベル間隔
ccycle=  5,                 udpack の icycle と同じ (cont(2)<0 のとき)
range=   -999.000  -999.000, コンターの最小，最大
cidx=  1  3,                コンターのラインインデックス 0 なら描かない
clabel=  T,                 コンターラベルを付けるや否や
tone=   21*-999.000,        トーンの境界値
pat=  20*-1,                トーン番号
map=  0,                    海岸線のラインタイプ 0 なら描かない
mapidx=  1,                 海岸線のラインインデックス
lay=  1,                    レイアウト形式番号 ( gglay1 - gglay3 に対応 )
wsn=  1,                    ワークステーション番号
mono= F,                    -mono  とするとモノクロ (iws=1)
print=F,                    -print とすると PS 出力 (iws=3)
str=  1,                    (時) 系列データの切り出しはじめ
end=  9999999,              (時) 系列データの切り出し終わり
step= 1,                    (時) 系列データの切り出し間隔
exch=  F,                   X 軸と Y 軸を入れ換える
tlnum= 6,                   トーンスケールのラベルを打つ数
softf=F,                    -softf でソフトフィル
color=0,                    カラー塗りわけの色数
cpat=18999,-99999,          カラー塗りわけの色の始め，間隔 (トーンパタン番号)
```

```
pixel=-1,-1,                  sgtonf による分割数．0 のときは sgtong を使う．
item='                ',      表示する item 文字列
unit='                ',      表示する unit 文字列
title= '                                ', 表示する title 文字列
dset='                ',      表示する dset 文字列
edit='                ',      表示に加える edit 文字列
ettl='                ',      表示に加える ettl 文字列
greset=  F                    データ内の描画情報設定（divs など）を無視するや否や
&end
```

-999 は不定の意味で, 適当なスケーリングがおこなわれる．x=y=z=-1 のときは, z=0 として扱われる．
ファイル名を指定しない場合は, 'gtool.out' が仮定される．
例：

```
% gtcont rain cont=200 tone=200,400,800 pat=2999,3999 tilte:Rain
```

以下, オプションの説明は, gtcont と違うもののみ示す．

8.3.2　gtcurv：折れ線描画

```
&option
x=  -1,
y=  -1,
z=  -1,
div=   -999.000   -999.000,  値の軸の目盛間隔，ラベル間隔
range=   -999.000 -999.000,
overlay=  1,                  折れ線を重ね合わせる数
lidx=  2  23*-1,              各折れ線のラインインデックス
ltype=  1  2  3  4 20*-1,     各折れ線のラインタイプ
bitlen=   -1.00000,           点線の間隔
ltitem='ITEM            ',    下部のラインタイトルのラベルとして何を使うか
lint=  T,                     下部のラインタイトルを描くや否や
lay=  1,
wsn=  1,
tick=  T,                     軸に格子の目盛を付けるやいなや
attl=  T,                     軸にタイトルを付けるや否や
str=  1,
end=  9999999,
step= 1,
exch=  F,                     T なら，座標を Y 軸に，データを X 軸にとる
stype=  0,                    値の軸の描画タイプ（log か，上向きか?）
item='                ',
unit='                ',
title='                                ',
```

```
dset='                ',
edit='                ',
ettl='                ',
greset=  F
&end
```

x=y=z=-1 のときは, x=0,z=0 が仮定される. overlay ¿1 のときは, 順次読み込まれたデータが overlay 本重ね合わされて描かれ, そのとき lidx, ltype が順次使われる.
ファイルは複数 (10 まで) 指定できる. そのときは, overlay ¿= ファイルの数となるよう, overlay が変更される.
lidx,ltype が ¡0 の線は, サイクリックな指定として扱う. すなわち, ltype = 1,2,3,4 のときには, ltype(5)=1,ltype(6)=2,... とみなす.
例 :

```
% gtcurv T.case1 T.case2  x=0 z=1 range=250,310 ltitem='DSET' -print
```

8.3.3　gtshow : データ・ヘッダのキャラクタ表示

```
&option
xstr=  1,                    x 軸の表示始め
xend=  999999,                       終わり
xstep=  1,                           間隔
x=  -999,                    x 軸の表示点 (-999 以外の時は上の 3 つの指定は無視)
ystr=  1,                    y 軸
yend=  999999,
ystep=  1,
y=  -999,
zstr=  1,                    z 軸
zend=  999999,
zstep=  1,
z=  -999,
tstr=  1,                    時間 (系列) 軸
tend=  999999,
tstep=  1,
t=  -999,
head=  F,                    -head でヘッダだけ表示
all=  F,                     -all  で値が空白のヘッダも表示
&end
```

8.3.4　gtset : データの線形変換

```
&option
ofs=  0.,                     オフセット値
```

```
fact=    1.00000,              ファクター値　出力は ofs+fact*data
out='gtool.out'
apend=F,                       ファイルへの付加出力
item='               ',
unit='               ',
title='                              ',
dset='               ',
edit='SET            ',
ettl='<gtset>        ',
divs=   -999.,                 描画パラメータ divs
divl=   -999.,                 描画パラメータ divl
dmin=   -999.,                 描画パラメータ dmin
-dmax=   -999.,                 描画パラメータ dmax
dmiss=  -999.,                 欠損値
styp=   -999,                  軸の書き方 (1:通常, 2:log, 負は下 (左) が正)
time=   -999,                  時刻
tdur=  -999,                   代表時間
utim='NUL            ',        時刻単位
date='NUL            ',        日付
aitm1='NUL            ',       軸名称 1
aitm2='NUL            ',       軸名称 2
aitm3='NUL            ',       軸名称 3
roptn=  -999.,                 オプションパラメータ
ioptn=  -999,                  オプションパラメータ
coptn='               ',       オプションパラメータ
greset=  F
&end
```

例：

```
% gtset rain fact=0.03456 unit:'mm/day' out:rain.mm
```

8.3.5　gtcon：データに一定値をセット

```
&option
val=  1.00000,                 値
out='gtool.out'
apend=F,
item='               ',
unit='               ',
title='                              ',
dset='               ',
edit='CON            ',
ettl='<gtcon>        ',
```

```
greset=  T
&end
```

例：

```
% gtcon T val=300. out:T.300 title:'reference temp.'
```

8.3.6　gtadd：二つのデータの和

```
&option
ofs1=  0.,                  オフセット値 (1)
ofs2=  0.,                  オフセット値 (2)
fact1=    1.00000,          ファクター値 (1)
fact2=    1.00000,          ファクター値 (2)
out='gtool.out          ',
apend=F,
item='                ',
unit='                ',
title='                               ',
dset='                ',
edit='ADD             ',
ettl='<gtadd>         ',
greset=  T
&end
```

出力は (ofs1+fact1*data1) + (ofs2+fact2*data2). ファイルが一つのみ指定された時は, もう一方として gtool.out を仮定.
同様なものとして, gtsub(差), gtmlt(積), gtdiv(商) がある.
例：

```
% gtadd Qrads Qradl out:Qradt item:QRADT title:'Total Radiative Heating'
```

8.3.7　gtsel：データの切りだし

```
&option
x=  -1,
y=  -1,
z=  -1,
str=  1,
end=  9999999,
step= 1,
sel= '                ',        選択する ITEM
out= 'gtool.out       '         出力ファイル名
```

```
apend=F,
item='               ',
unit='               ',
title='                                ',
dset='               ',
edit='               ',
ettl='               ',
divs=   -999.,
divl=   -999.,
dmin=   -999.,
dmax=   -999.,
dmiss=  -999.,
styp=   -999,
roptn=  -999.,
ioptn=  -999,
coptn='               ',
greset=  F,
&end
```

x=y=z=-1 の場合はデータ本体には何も加工しない.
例：

```
% gtsel T z=1 str=50 end=100 out:Ta title:'Surface Air Temp.' item:Ta
```

8.3.8　gtext：データの範囲切り出し, 拡張

```
&option
x='               ',          新しい x 軸の名称
y='               ',          新しい y 軸の名称
z='               ',          新しい z 軸の名称
xstr=  0,                     x 軸の範囲はじめ
xend=  0,                     x 軸の範囲終わり
ystr=  0,                     y 軸の範囲はじめ
yend=  0,                     y 軸の範囲終わり
zstr=  0,                     z 軸の範囲はじめ
zend=  0,                     z 軸の範囲終わり
ref='             ',          軸の構成を指定したファイルと同じとする
out='gtool.out    ',
apend=F
&end
```

例：

```
% gtsel T zstr=1 zend=5 out:T.bl
```

```
% gtsel Ps z:GSIG16.P out:Ps.3d
% gtsel Ps ref:T out:Ps.3d  （ Ps を 3 次元データにする ）
```

8.3.9　gtavr : 時間 (系列) 平均

```
&option
wright=  F,                    TDUR の重みを付けるや否や
str=  1,
end=  9999999,
step= 1,
out='gtool.out     '
apend=F,
item='               ',
unit='               ',
title='                              ',
dset='               ',
edit='               ',
ettl='               ',
greset=  F
&end
```

連続するデータ (通常は時系列) の平均を作成する
例:

```
% gtavr rain out:rain.av
```

8.3.10　gtseq : 時系列の作成

```
&option
out='gtool.out     '
apend=F,
axname='              ',           軸の名称
axitem='              ',           軸目盛の値として使用する欄名
item='               ',
unit='               ',
title='                              ',
dset='               ',
edit='               ',
ettl='               ',
greset=  F
&end
```

系列データから, 1 次元大きい単一データを作成. デフォルトでは時刻が目盛として扱われ, 時間軸の軸ファイルが出力される. 領域あふれに注意.

```
% gtseq rain.y32 out:rain.y32.seq
```

8.4　コマンドの組み合わせの例

- 例 1 : x-t 図のプロット

```
% gtsel rain y=32 out:rain.y32
% gtseq rain.y.32 out:rain.y32.seq
% gtcont rain.y32.seq
```

これは, 以下のようにもできる.

```
% gtsel rain y=32 out:rain.y32
% gtseq rain.y.32 && gtcont
```

- 例 2: 可降水量の算出

```
% gtset Ps fact=10.20 && gtext ref:q out:Ps.3d unit:'kg/m**2'
% gtmul Ps.3d q && gtsel z=0  out:PW title:'Preciptable Water' item:PW
```

第9章　参考文献

[1] 地球流体電脳ライブラリ (dcl-5.0) 利用の手引き, 地球流体電脳倶楽部 (1995)

索 引

GFAOPN, 32
GFAOPQ, 32
GFCLSE, 31
GFOOPN, 32
GFOOPQ, 33
GFOPEN, 31
GFQADJ, 34
GFRDHD, 29
GFRDSL, 28
GFRDTS, 34
GFREAD, 28
GFREDZ, 29
GFROPN, 30
GFROPQ, 32
GFSADJ, 33
GFWOPN, 30
GFWOPQ, 32
GFWRIT, 29
GFWRTZ, 30
GFWTTS, 34
GGAXES, 53
GGAXIS, 54
GGAXIX, 63
GGAXRR, 54
GGAXRS, 63
GGAXSZ, 62
GGCDSC, 62
GGCGET, 61
GGCLSE, 52
GGCNTR, 54
GGCSET, 61
GGCTON, 55
GGCURV, 57
GGLAY1, 53
GGLINT, 58
GGLTRS, 64
GGMAPS, 55
GGMARK, 58
GGOPEN, 52
GGPDSC, 62
GGPGET, 61
GGPSET, 61
GGRANG, 59
GGSCAT, 57
GGSTON, 56
GGTONE, 55
GGVECT, 56
GHCGET, 20
GHCGTS, 21
GHCOPY, 22
GHCSET, 21
GHCSQ1, 26
GHCSQ2, 27
GHCSTS, 22
GHEADD, 22
GHERST, 23
GHESET, 23
GHNINQ, 27
GHPACA, 26
GHPACK, 23
GHPCLR, 27
GHPGET, 20
GHPINQ, 21
GHPSET, 20
GHPTRN, 27
GHQENM, 23
GHRSGP, 27
GHUPAC, 25
GPCAL1, 48
GPCAL2, 49
GPCAL3, 49
GPCAL4, 50
GPFADD, 45
GPFCON, 44

GPFDIV, 47
GPFFCT, 44
GPFFNA, 45
GPFFNB, 47
GPFINC, 43
GPFMLT, 46
GPFSET, 44
GPFSUB, 46
GPTAVG, 41
GPTAVO, 42
GPTAVR, 42
GPTSEQ, 42
GPTSQN, 43
GPTSQR, 43
GPXAVG, 37
GPXCMI, 39
GPXCOM, 38
GPXEDY, 38
GPXEXC, 40
GPXEXP, 39
GPXEXT, 40
GPXRDC, 37
GPXSEL, 36
gtadd, 81
gtavr, 83
GTCGET, 18
gtcon, 80
gtcont, 77
GTCSET, 18
gtcurv, 78
gtext, 82
GTPGET, 17
GTPSET, 17
gtsel, 81
gtseq, 83
gtset, 79
gtshow, 79
GTSIGN, 19
GTSIZE, 18
GUAIAH, 69
GUAIAX, 69
GUAXCK, 66
GUCSIZ, 65
GUCTIM, 72
GUCTXT, 73
GUEAXN, 68
GUQAXD, 67
GUQAXS, 67
GUQAXV, 66
GUQIAH, 70
GUQIAX, 70
GUQNOW, 72
GUQSIZ, 65
GUQTIM, 71
GUSIAX, 68
GUSMIS, 71
GUSNOW, 72
GUSZCK, 71
GUSZCZ, 71
GUTPAC, 73
GUTUPC, 73
GUVINT, 74

PORAXZ, 63

サイズチェック機能, 14
スケーリングタイプ, 10
データユニット, 10
データレコード, 10
データ記述子, 10
ファイル格子情報, 12
ヘッダー, 7
ヘッダーの各記述子の説明, 9
ヘッダーレコード, 10
モジュールの使用法, 16

仮引数リストの継続行の記号, 15

格子位置情報, 11
格子重み情報, 11
格子情報, 11
格子情報のファイル名, 13
格子情報のヘッダー, 12
格子情報の種類の略称, 12
格子番号, 12

記述子, 10

識別記述子, 10
周期的な座標軸, 12

属性記述子, 10

電脳ライブラリ, 14

内部設定格子情報, 12

標準ファイル形式, 10

編集記述子, 10