

平成17年9月30日  
気象庁予報部  
気象庁地球環境・海洋部

## 配信資料に関する技術情報(気象編)第205号

～平成18年3月からの数値予報モデルGPV等の変更について～

当庁では、平成18年3月の数値予報計算機(NAPS)の更新にあわせて、メソ数値予報モデル、週間アンサンブル数値予報モデル、1か月予報アンサンブルモデルの改善を行う計画です。これに伴い、以下の通り同モデルGPV等の配信内容を変更します。

### 1. メソ数値予報モデルGPV、週間アンサンブル数値予報モデルGPVの変更

メソ数値予報モデル(MSM)は、平成18年3月から計算の格子間隔を現在の10kmメッシュから5kmメッシュに細分化するとともに、運用回数を現在の1日4回から8回とします。当面、予報時間については15時間としますが、平成19年には1日4回(03、09、15、21UTC)33時間に延長する予定です。

週間アンサンブル数値予報モデルは、平成18年3月から予報メンバー数を現在の25から51に増やすとともに、提供プロダクトを拡充します。

詳細については、解説資料1をご覧ください。

### 2. 1か月予報アンサンブル格子点値および1か月予報ガイダンスの変更

1か月予報アンサンブルモデルについて、平成18年3月からメンバー数や出力要素を増やす改善を行うとともに、1か月予報アンサンブル格子点値と1か月予報ガイダンスの配信内容を拡充します。詳細については、解説資料2をご覧ください。

平成 17 年 9 月  
気象庁予報部

## メソ数値予報モデル GPV、週間アンサンブル数値予報モデル GPV の変更

数値予報計算機（NAPS）の更新に伴うメソ数値予報モデル及び週間アンサンブル数値予報モデル改善にあわせて、平成 18 年 3 月にこれらの数値予報モデル GPV の配信内容を変更します。

### 1. 数値予報モデル GPV の変更内容

メソ数値予報モデル（MSM）は、平成 18 年 3 月から計算の格子間隔を現在の 10km メッシュから 5km メッシュに細分化するとともに、運用回数を現在の 1 日 4 回から 8 回とします。当面、予報時間については 15 時間としますが、平成 19 年には 1 日 4 回（03、09、15、21UTC）33 時間に延長する予定です。

週間アンサンブル数値予報モデルは、平成 18 年 3 月から予報メンバー数を現在の 25 から 51 に増やすとともに、提供プロダクトを拡充します。

これらの改善に伴い GPV（ファイル形式）の配信内容を添付資料 1-1、1-2 のとおり変更します。

### 2. 提供開始日及び提供時刻

平成 18 年 3 月 1 日（水）00 時（協定世界時）初期値から配信を予定しています。なお、提供開始に先立ち試験配信を行う予定ですが、試験配信開始日については別途お知らせします。

気象業務支援センターへの送信完了時刻は、原則として以下のとおりです。

- ・メソ数値予報モデル GPV（00,06,12,18UTC 初期値） 初期時刻 + 2 時間 10 分
- ・メソ数値予報モデル GPV（03,09,15,21UTC 初期値） 初期時刻 + 2 時間 30 分
- ・週間アンサンブル数値予報モデル GPV（12UTC 初期値） 2040UTC

但し、メソ数値予報モデル GPV について、障害等により初期時刻から 4 時間以内に配信を完了できない場合は、当該時刻のファイルの配信を中止します。

### 3. 数値予報モデル GPV のファイル名

#### (1) メソ数値予報モデル GPV

（18 年 3 月から）

- ・地上、15 時間予報データ

Z\_C\_RJTD\_yyyyMMddhhmmss\_MSM\_GP\_V\_Rjp\_Lsurf\_FH00-15\_grib2.bin

- ・気圧面、15 時間予報データ

Z\_C\_RJTD\_yyyyMMddhhmmss\_MSM\_GP\_V\_Rjp\_L-pall\_FH00-15\_grib2.bin

( 19 年から )

- ・ 地上、15 時間予報データ

Z\_C\_RJTD\_yyyyMMddhhmmss\_MSM\_GPV\_Rjp\_Lsurf\_FH00-15\_grib2.bin

- ・ 気圧面、15 時間予報データ

Z\_C\_RJTD\_yyyyMMddhhmmss\_MSM\_GPV\_Rjp\_L-pall\_FH00-15\_grib2.bin

- ・ 地上、33 時間予報データ

Z\_C\_RJTD\_yyyyMMddhhmmss\_MSM\_GPV\_Rjp\_Lsurf\_FH00-33\_grib2.bin

- ・ 気圧面、33 時間予報データ

Z\_C\_RJTD\_yyyyMMddhhmmss\_MSM\_GPV\_Rjp\_L-pall\_FH00-33\_grib2.bin

(2) 週間アンサンブル数値予報モデル GPV

- ・ 全球データ

Z\_C\_RJTD\_yyyyMMddhhmmss\_EPSW\_GPV\_Rgl\_FD00-08\_grib2.bin

- ・ 日本域データ

Z\_C\_RJTD\_yyyyMMddhhmmss\_EPSW\_GPV\_Rjp\_FD00-08\_grib2.bin

Z と C の間にはアンダースコアが 2 個、その他のアンダースコアは 1 個。  
yyyyMMddhhmmss はデータの初期時刻の年月日時分秒を UTC (協定世界時) で設定。

#### 4. ファイル形式

今回提供を開始するデータのファイル形式は、全て国際気象通報式 FM92 GRIB 二進形式格子点資料気象通報式(第 2 版)(以下、「GRIB2」という)に則っています。GRIB2 の詳細については国際気象通報式・別冊に詳しく記述されていますので、当該資料を参照願います。また、今回提供を開始するデータの GRIB2 各節の詳細は、添付資料 2-1、2-2 を参照してください。

また、当該データの解読(デコード)処理については、添付資料 3 を参照してください。

以下のサンプルデータを気象業務支援センターから提供しますので、必要な場合はご利用下さい。

(メソ数値予報モデル GPV)

- ・ 地上及び気圧面の 15 時間予報(03UTC 初期値)

- ・ 地上及び気圧面の 33 時間予報(03UTC 初期値)

サンプルデータでは、33 時間予報の 16-30(31-33) 時間予報には、1-15(1-3) 時間予報と日時を除いて同じデータを格納しています。

(週間アンサンブル数値予報モデル GPV)

- ・ 全球及び日本域 192 時間予報(12UTC 初期値)

## メソ数値予報モデルG P V

## ( 1 ) 概要

- 初期値 : 00, 03, 06, 09, 12, 15, 18, 21UTC ( 1 日 8 回 )
- 予報時間 : 15 時間予報、地上は 1 時間間隔、気圧面は 3 時間間隔  
 なお、平成 19 年度に 03,09,15,21UTC は 33 時間予報とする。
- 格子系 : 等緯度等経度
- 格子間隔 : 地上は緯度 0.05 度 × 経度 0.0625 度 ( 格子数 481 × 505 )  
 気圧面は緯度 0.1 度 × 経度 0.125 度 ( 格子数 241 × 253 )
- 領域 : ( 47.6N, 120E ) を北西端、( 22.4N, 150E ) を南東端とする領域
- データ量 : 約 114MB/回 × 4 回、約 237MB/回 × 4 回 = 1,404MB/日  
 ( 33 時間予報開始後のデータ量 )
- フォーマット : GRIB2

## ( 2 ) データ内容

## 地上物理量

	海面更正 気圧	地上 気圧	風	気温	相対 湿度	時間 降水量	雲量
地上							

## 気圧面物理量

気圧面 (hPa)	高度	風	気温	上昇流	相対 湿度
1000					
975					
950					
925					
900					
850					
800					
700					
600					
500					
400					
300					
250					
200					
150					
100					

は 2 要素分のデータ ( 風の場合、東西方向と南北方向の 2 要素 )

は 4 要素分のデータ ( 雲量の場合、全雲量、上層雲量、中層雲量、下層雲量の 4 要素 )

## 週間アンサンブル数値予報モデル G P V

## 1. 概要

## (1) 全球

初期値 : 12UTC  
 予報時間 : 192 時間予報, 12 時間間隔  
 アンサンブルメンバ数 : 51 メンバ  
 格子系 : 等緯度等経度  
 格子間隔 : 2.5 度×2.5 度 (格子数 144×73)  
 領域 : 全球  
 データ量 : 約 288MB/回  
 フォーマット : GRIB2

## (2) 日本域

初期値 : 12UTC  
 予報時間 : 192 時間予報, 6 時間間隔  
 アンサンブルメンバ数 : 51 メンバ  
 格子系 : 等緯度等経度  
 格子間隔 : 1.25 度×1.25 度 (格子数 73×40)  
 領域 : 日本域(北西端 71.25N, 90E, 南東端 22.5N, 180E の矩形領域)  
 データ量 : 約 172MB/回  
 フォーマット : GRIB2

## 2. データ内容

## (1) 全球

通報面	高度	風	気温	相対湿度	積算降水量	全雲量	海面更正気圧	地上気圧
地上								
850hPa								
500hPa								
300hPa								

## (2) 日本域

通報面	高度	風	気温	相対湿度	積算降水量	上昇流	全雲量	海面更正気圧	地上気圧
地上									
925hPa									
850hPa									
700hPa									
500hPa									

は2要素分のデータ(風の場合, 東西方向と南北方向の2要素)

# GRIB2通報式による メソ数值予報モデル格子点値 データフォーマット

平成17年 9月

気象庁予報部

## 1. データについて

- ・フォーマットは、国際気象通報式FM92GRIB 二進形式格子点資料気象通報式(第2版)(以下、「GRIB2」という)に則っている。
- ・地上物理量を含むファイルと、気圧面物理量を含むファイルに分かれており、格子数、格子間隔、時間間隔なども異なる。
- ・第4節(プロダクト定義節)で用いるテンプレートは、時間降水量のみテンプレート4.8を用い、他の物理量はテンプレート4.0を用いる。
- ・要素、水平面が現れる順序は不定である。
- ・GRIB2中の作成ステータスを利用して試験を行う場合があるので、必ず作成ステータス(第1節第20オクテット)を参照すること。

以下は、GRIB2 に共通である。

- ・各フォーマット中のバイナリデータは、ビッグエンディアンである。
- ・負の値は最上位ビットを1にすることにより示す(2の補数表現ではない)
- ・単純圧縮において元のデータYは、次の式で復元できる。

$$Y = (R + X \times 2^E) \div 10^D$$

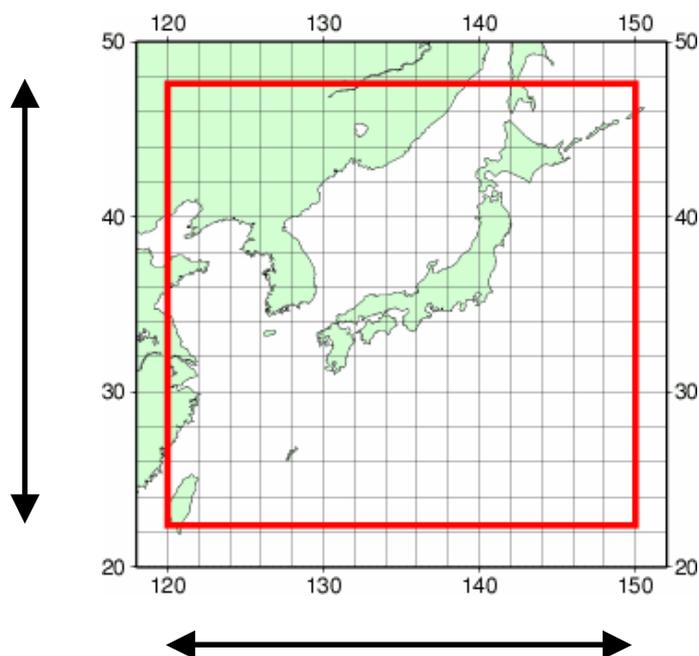
E = 二進尺度因子

D = 十進尺度因子

R = 参照値

X = 圧縮された値

### データの範囲



2.1 メソ数値予報モデルの地上物理量に用いるGRIB2のフォーマットおよびテンプレートの詳細

節番号	節の名称 該当テンプレート	オクテット	内容	表	値	備考				
第0節	指示節	1~4	GRIB			"GRIB" 国際アルファベットNo.5 (CCITT IA5)				
		5~6	保留			missing				
		7	資料分野		符号表0.0		0	気象分野		
		8	GRIB版番号				2			
		9~16	GRIB報全体の長さ				*****	63,774,673 (15時間予報) 135,931,057 (33時間予報)		
		1~4	節の長さ				21			
		5	節番号				1			
		6~7	作成中枢の識別		共通符号表C-1		34	東京		
		8~9	作成副中枢				0			
		10	GRIBマスター表バージョン番号		符号表1.0		2	現行運用バージョン番号		
第1節	識別節	11	GRIB地域表バージョン番号		符号表1.1	1	地域表バージョン1			
		12	参照時刻の意味		符号表1.2	1	予報の開始時刻			
		13~14	資料の参照時刻(年)				*****			
		15	資料の参照時刻(月)				*****			
		16	資料の参照時刻(日)				*****			
		17	資料の参照時刻(時)				*****			
		18	資料の参照時刻(分)				*****			
		19	資料の参照時刻(秒)				*****			
		20	作成ステータス		符号表1.3		T	0=現業プロダクト、1=現業的試験プロダクト		
		21	資料の種類		符号表1.4		1	予報プロダクト		
		第2節	地域使用節	不使用				省略		
		第3節	格子系定義節	1~4	節の長さ			72		
				5	節番号				3	
				6	格子系定義の出典		符号表3.0		0	符号表3.1参照
				7~10	資料点数				242905	505 x 481
11	格子点数を定義するリストのオクテット数						0			
12	格子点数を定義するリストの説明						0			
13~14	格子系定義テンプレート番号				符号表3.1		0	緯度・経度格子		
15	地球の形状				符号表3.2		6	半径6,371kmの球体と仮定した地球		
16	地球球体の半径の尺度因子						missing			
17~20	地球球体の尺度付き半径						missing			
21	地球回転楕円体の長軸の尺度因子						missing			
22~25	地球回転楕円体の長軸の尺度付きの長さ						missing			
26	地球回転楕円体の短軸の尺度因子						missing			
27~30	地球回転楕円体の短軸の尺度付きの長さ						missing			
31~34	緯線に沿った格子点数						481			
35~38	経線に沿った格子点数						505			
39~42	原作領域の基本角						0			
43~46	端点の経度及び緯度並びに方向増分の定義に使われる基本角の細分						missing			
47~50	最初の格子点の緯度				10**-6度単位		47600000	北緯47.6度		
51~54	最初の格子点の経度				10**-6度単位		120000000	東経120度		
55	分解能及び成分フラグ				フラグ表3.3		0x30			
56~59	最後の格子点の緯度				10**-6度単位		22400000	北緯22.4度		
60~63	最後の格子点の経度				10**-6度単位		150000000	東経150度		
64~67	方向の増分				10**-6度単位		62500	0.0625度		
68~71	方向の増分				10**-6度単位		50000	0.05度		
72	走査モード				フラグ表3.4		0x00			
第4節	プロダクト定義節			1~4	節の長さ			*****	34 または 58	
				5	節番号				4	
				6~7	テンプレート直後の座標値の数				0	
				8~9	プロダクト定義テンプレート番号		符号表4.0		*****	0=ある時刻の、ある水平面における予報、8=連続又は不連続な時間間隔の水平面における統計値
				10	パラメータカテゴリー		符号表4.1		1	
				11	パラメータ番号		符号表4.2		1	
				12	作成処理の種類		符号表4.3		*****	1=初期化 2=予報
				13	背景作成処理識別符		JMA定義		*****	31=メソ数値予報(数値予報モデルの改良により変更される場合がある)
				14	解析又は予報の作成処理識別符				missing	
				15~16	観測資料の参照時刻からの締切時間(時)				0	
				17	観測資料の参照時刻からの締切時間(分)				50	
				18	期間の単位の指示符		符号表4.4		1	時
				19~22	予報時間				3	
				23	第一固定面の種類		符号表4.5		2	
				24	第一固定面の尺度因子				2	
				25~28	第一固定面の尺度付きの値				2	
				29	第二固定面の種類		符号表4.5		missing	
				30	第二固定面の尺度因子				missing	
				31~34	第二固定面の尺度付きの値				missing	
				35~36	全時間間隔の終了時(年)				3	
				37	全時間間隔の終了時(月)				3	
		38	全時間間隔の終了時(日)				3			
		39	全時間間隔の終了時(時)				3			
		40	全時間間隔の終了時(分)				3			
		41	全時間間隔の終了時(秒)				3			
		42	統計を算出するために使用した時間間隔を記述する期間の仕様の数				1			
		43~46	統計処理における欠測資料の総数				0			
		47	統計処理の種類				1			
		48	統計処理の時間増分の種類				2			
		49	統計処理の時間の単位の指示符				1			
		50~53	統計処理した期間の長さ				1			
		54	連続的な資料場間の増分に関する時間の単位の指示符				1			
		55~58	連続的な資料場間の時間の増分				0			
		第5節	資料表現節	1~4	節の長さ			21		
				5	節番号				5	
				6~9	全資料点数				242905	505 x 481
				10~11	資料表現テンプレート番号		符号表5.0		0	格子点資料・単純圧縮
				12~15	参照値(R) (IEEE 32ビット浮動小数点)				R	Rは可変
				16~17	二進尺度因子(E)				E	Eは可変
				18~19	十進尺度因子(D)				D	Dは可変
				20	単純圧縮による各圧縮値のビット数				12	
				21	原資料場の値の種類		符号表5.1		0	浮動小数点
				第6節	ビットマップ節	1~4	節の長さ			6
				5	節番号			6		
				6	ビットマップ指示符			255	ビットマップを適用せず	
		第7節	資料節	1~4	節の長さ			364363		
				5	節番号				7	
6~nn	単純圧縮オクテット列						X-	単純圧縮された格子点値の列		
第8節	終端節	1~4	7777			"7777" 国際アルファベットNo.5 (CCITT IA5)				

(注) 値が"missing"の場合、そのデータは全ビット1の値、英数字の変数名や"\*\*\*\*\*"は可変を示す。

要素および水平面毎に、第1節～第7節を繰り返す

2.2 メソ数値予報モデルの気圧面物理量に用いるGRIB2のフォーマットおよびテンプレートの詳細

節番号	節の名称・ 該当テンプレート	オクテット	内容	表	値	備考		
第0節	指示節	1~4	GRIB		"GRIB"	国際アルファベットNo.5 (CCITT IA5)		
		5~6	保留		missing			
		7	資料分野	符号表0.0	0	0	気象分野	
		8	GRIB版番号			2		
		9~16	GRIB報全体の長さ		*****	50,522,465 (15時間予報) 101,044,817 (33時間予報)		
		第1節	識別節	1~4	節の長さ		21	
				5	節番号		1	
				6~7	作成中枢の識別	共通符号表C-1	34	東京
				8~9	作成副中枢		0	
				10	GRIBマスター表バージョン番号	符号表1.0	0	現行運用バージョン番号
11	GRIB地域表バージョン番号			符号表1.1	1	地域表バージョン1		
12	参照時刻の意味			符号表1.2	1	予報の開始時刻		
13~14	資料の参照時刻(年)				*****			
15	資料の参照時刻(月)				*****			
16	資料の参照時刻(日)				*****			
17	資料の参照時刻(時)				*****			
18	資料の参照時刻(分)				*****			
19	資料の参照時刻(秒)				*****			
20	作成ステータス	符号表1.3	T	0=現業プロダクト, 1=現業的試験プロダクト				
21	資料の種類	符号表1.4	1	予報プロダクト				
第2節	地域使用節	不使用			省略			
第3節	格子系定義節	1~4	節の長さ		72			
		5	節番号		3			
		6	格子系定義の出典	符号表3.0	0	符号表3.1参照		
		7~10	資料点数		60973	253x241		
		11	格子点数を定義するリストのオクテット数		0			
		12	格子点数を定義するリストの説明		0			
		13~14	格子系定義テンプレート番号	符号表3.1	0	緯度・経度格子		
		15	地球の形状	符号表3.2	6	半径6,371kmの球体と仮定した地球		
		16	地球球体の半径の尺度因子		missing			
		17~20	地球球体の尺度付き半径		missing			
		21	地球回転楕円体の長軸の尺度因子		missing			
		22~25	地球回転楕円体の長軸の尺度付きの長さ		missing			
		26	地球回転楕円体の短軸の尺度因子		missing			
		27~30	地球回転楕円体の短軸の尺度付きの長さ		missing			
		31~34	経線に沿った格子点数		241			
		35~38	経線に沿った格子点数		253			
		39~42	原作成領域の基本角		0			
		43~46	端点の経度及び緯度並びに方向増分の定義に使われる基本角の細分		missing			
		47~50	最初の格子点の緯度	10*-6度単位	47600000	北緯47.6度		
		51~54	最初の格子点の経度	10*-6度単位	120000000	東経120度		
		55	分解能及び成分フラグ	フラグ表3.3	0x30			
		56~59	最後の格子点の緯度	10*-6度単位	22400000	北緯22.4度		
		60~63	最後の格子点の経度	10*-6度単位	150000000	東経150度		
		64~67	方向の増分	10*-6度単位	125000	0.125度		
		68~71	方向の増分	10*-6度単位	100000	0.1度		
		72	走査モード	フラグ表3.4	0x00			
		第4節	プロダクト定義節	1~4	節の長さ		34	
				5	節番号		4	
				6~7	テンプレート直後の座標値の数		0	
				8~9	プロダクト定義テンプレート番号	符号表4.0	0	ある時刻の、ある水平面における解析
				10	パラメータカテゴリー	符号表4.1	1	
				11	パラメータ番号	符号表4.2	1	
				12	作成処理の種類	符号表4.3	*****	1=初期化 2=予報
13	背景作成処理識別符			JMA定義	*****	31=メソ数値予報(数値予報モデルの改良により変更される場合がある)		
14	解析又は予報の作成処理識別符				missing			
15~16	観測資料の参照時刻からの締切時間(時)				0			
17	観測資料の参照時刻からの締切時間(分)				50			
18	期間の単位の指示符			符号表4.4	1	1時		
19~22	予報時間				3			
23	第一固定面の種類			符号表4.5	2			
24	第一固定面の尺度因子				2			
25~28	第一固定面の尺度付きの値				2			
29	第二固定面の種類			符号表4.5	missing			
30	第二固定面の尺度因子				missing			
31~34	第二固定面の尺度付きの値				missing			
第5節	資料表現節			1~4	節の長さ		21	
				5	節番号		5	
		6~9	全資料点数		60973	253x241		
		10~11	資料表現テンプレート番号	符号表5.0	0	格子点資料 - 単純圧縮		
		12~15	参照値(R) (IEEE 32ビット浮動小数点)		R	Rは可変		
		16~17	二進尺度因子(E)		E	Eは可変		
		18~19	十進尺度因子(D)		D	Dは可変		
		20	単純圧縮による各圧縮値のビット数		12			
		21	原資料場の値の種類	符号表5.1	0	浮動小数点		
		第6節	ビットマップ節	1~4	節の長さ		6	
5	節番号				6			
6	ビットマップ指示符				255	ビットマップを適用せず		
7	資料節				91465			
第7節	資料節	1~4	節の長さ		7			
		5	節番号		7			
		6~nn	単純圧縮オクテット列		X~	単純圧縮された格子点値の列		
第8節	終端節	1~4	7777		"7777" 国際アルファベットNo.5 (CCITT IA5)			

要素および水平面毎に、第4節~第7節を繰り返す

(注) 値が「missing」の場合、そのデータは全ビット1の値、英数字の変数名や「\*\*\*\*\*」は可変を示す。

1 要素の表現 (第4節 10~11オクテットについて)

	10オクテット パラメータカテゴリ (符号表4.1)	11オクテット パラメータ番号 (符号表4.2)
気温	0 (温度)	0 (温度 K)
相対湿度	1 (湿度)	1 (相対湿度 %)
毎時降水量	"	8 (総降水量 $\text{kg}\cdot\text{m}^{-2}$ )
風の東西成分	2 (運動量)	2 (風のu成分 $\text{m}/\text{s}$ )
風の南北成分	"	3 (風のv成分 $\text{m}/\text{s}$ )
上昇流	"	8 (鉛直速度(気圧) $\text{Pa}/\text{s}$ )
地上気圧	3 (質量)	0 (気圧 Pa)
海面更正気圧	"	1 (海面更正気圧 Pa)
高度	"	5 (ジオポテンシャル高度 gpm)
全雲量	6 (雲)	1 (全雲量 %)
下層雲量	"	3 (下層雲量 %)
中層雲量	"	4 (中層雲量 %)
上層雲量	"	5 (上層雲量 %)

2 固定面の表現 (第4節 23~28オクテットについて)

	23オクテット 第一固定面の種類 (符号表4.5)	24オクテット 第一固定面の 尺度因子	25~28オクテット 第一固定面の 尺度付きの値
地面	1 (地面又は水面)	missing	missing
平均海面	101 (平均海面)	missing	missing
地上10m (風)	103 (地上からの特定高度面)	0	10
地上1.5m(気温,RH)	103 (地上からの特定高度面)	1	15
1000 hPa	100 (等圧面 Pa)	-2	1000
975 hPa	"	"	975
950 hPa	"	"	950
925 hPa	"	"	925
900 hPa	"	"	900
850 hPa	"	"	850
800 hPa	"	"	800
700 hPa	"	"	700
600 hPa	"	"	600
500 hPa	"	"	500
400 hPa	"	"	400
300 hPa	"	"	300
250 hPa	"	"	250
200 hPa	"	"	200
150 hPa	"	"	150
100 hPa	"	"	100

### 3 時刻の表現 (特に降水量について)

プロダクト定義節(第4節)は、要素が毎時降水量の場合は、テンプレート4.8、その他の要素ではテンプレート4.0を用いる。

テンプレート4.0の場合、参照時刻(第1節)に予報時間(第4節)を加えた時刻が資料節の内容になる。

テンプレート4.8 即ち降水量の場合、参照時刻(第1節)に予報時間(第4節)を加えた時刻から全期間の終了時(第4節)が示す時刻までの降水量が資料節の内容になる。

メソ数値予報モデルGPVにおいて降水量は1時間毎の値として表現される。

(2006年1月10日12UTCを初期値とする時間降水量の場合)

第1節	オクテット 13~19	参照時刻	2006.01.10.12:00		
第4節	18	期間の単位の 指示符	1	1	1
第4節	19~22	予報時間	0	1	2
第4節	35~41	全時間の終了	2006.01.10.13:00	2006.01.10.14:00	2006.01.10.15:00
第4節	50~53	統計処理した 期間の長さ	1	1	1

(単位は  
時間)

統計期間	開始時刻	+	2006.01.10.12:00	2006.01.10.13:00	2006.01.10.14:00
	終了時刻		2006.01.10.13:00	2006.01.10.14:00	2006.01.10.15:00
	資料節の内容		1時間目の 時間降水量	2時間目の 時間降水量	3時間目の 時間降水量

# GRIB2通報式による 週間アンサンブル数値予報モデル格子点値 データフォーマット

平成17年 9月

気象庁予報部

## 1 . データについて

- ・ フォーマットは、国際気象通報式FM92GRIB 二進形式格子点資料気象通報式(第2版)(以下、「GRIB2」という)に則っている。
- ・ 全球を範囲とするファイルと、日本域を範囲とするファイルに分かれており、要素の他、格子数、格子間隔、時間間隔なども異なる。
- ・ 第4節(プロダクト定義節)で用いるテンプレートは、積算降水量のみテンプレート4.11 を用い、他の物理量はテンプレート4.1を用いる。
- ・ メンバ、要素、水平面が現れる順序は不定である。
- ・ GRIB2中の作成ステータスを利用して試験を行う場合があるので、必ず作成ステータス(第1節第20オクテット)を参照すること。

以下は、GRIB2 に共通である。

- ・ 各フォーマット中のバイナリデータは、ビッグエンディアンである。
- ・ 負の値は最上位ビットを1にすることにより示す(2の補数表現ではない)
- ・ 単純圧縮において元のデータYは、次の式で復元できる。

$$Y = (R + X \times 2^E) \div 10^D$$

E = 二進尺度因子  
D = 十進尺度因子  
R = 参照値  
X = 圧縮された値

2.1 週間アンサンブル数値予報モデルの全球に用いるGRIB2のフォーマットおよびテンプレートの詳細

節番号	節の名称 該当テンプレート	オクテット	内容	表	値	備考				
第0節	指示節	1~4	GRIB		"GRIB"	国際アルファベットNo.5(CCITT IA5)				
		5~6	保留		missing					
		7	資料分野	符号表0.0	0	0	気象分野			
		8	GRIB版番号			2				
		9~16	GRIB報全体の長さ			287556269				
第1節	識別節	1~4	節の長さ			21				
		5	節番号			1				
		6~7	作成中枢の識別	共通符号表C-1	34	東京				
		8~9	作成副中枢			0				
		10	GRIBマスター表バージョン番号	符号表1.0	2	現行運用バージョン番号				
		11	GRIB地域表バージョン番号	符号表1.1	1	地域表バージョン1				
		12	参照時刻の意味	符号表1.2	1	予報の開始時刻				
		13~14	資料の参照時刻(年)			*****				
		15	資料の参照時刻(月)			*****				
		16	資料の参照時刻(日)			*****				
		17	資料の参照時刻(時)			*****				
		18	資料の参照時刻(分)			*****				
		19	資料の参照時刻(秒)			*****				
		20	作成ステータス	符号表1.3	T	0=現業プロダクト、1=現業的試験プロダクト				
		21	資料の種類	符号表1.4	5	コントロール及び摂動予報プロダクト				
		第2節	地域使用節	不使用				省略		
		第3節	格子系定義節	1~4	節の長さ			72		
				5	節番号			3		
				6	格子系定義の出典	符号表3.0	0	符号表3.1参照		
7~10	資料点数					10512	144x73			
11	格子点数を定義するリストのオクテット数					0				
12	格子点数を定義するリストの説明					0				
13~14	格子系定義テンプレート番号			符号表3.1	0	緯度・経度格子				
15	地球の形状			符号表3.2	6	半径6,371kmの球体と仮定した地球				
16	地球球体の半径の尺度因子					missing				
17~20	地球球体の尺度付き半径					missing				
21	地球回転楕円体の長軸の尺度因子					missing				
22~25	地球回転楕円体の長軸の尺度付きの長さ					missing				
26	地球回転楕円体の短軸の尺度因子					missing				
27~30	地球回転楕円体の短軸の尺度付きの長さ					missing				
31~34	緯線に沿った格子点数					144				
35~38	経線に沿った格子点数					73				
39~42	原作成領域の基本角					0				
43~46	端点の経度及び緯度並びに方向増分の定義に使われる基本角の細分					missing				
47~50	最初の格子点の緯度			10**-6度単位	90000000	北緯90.0度				
51~54	最初の格子点の経度			10**-6度単位	0	東経0度				
55	分解能及び成分フラグ			フラグ表3.3	0x30					
56~59	最後の格子点の緯度			10**-6度単位	-90000000	南緯90度				
60~63	最後の格子点の経度			10**-6度単位	357500000	東経357.5度				
64~67	方向の増分			10**-6度単位	2500000	2.5度				
68~71	方向の増分			10**-6度単位	2500000	2.5度				
72	走査モード			フラグ表3.4	0x00					
第4節	プロダクト定義節			1~4	節の長さ			*****		
				5	節番号			4		
				6~7	テンプレート直後の座標値の数			0		
				8~9	プロダクト定義テンプレート番号	符号表4.0	*****	1=ある時刻の、ある水平面における個々のアンサンブル予報、11=連続又は不連続な時間間隔の水平面における個々のアンサンブル		
				10	パラメータカテゴリー	符号表4.1	1			
				11	パラメータ番号	符号表4.2	1			
				12	作成処理の種類	符号表4.3	4	アンサンブル予報		
				13	背景作成処理識別符	JMA定義	*****	12=週間アンサンブル予報(数値予報モデルの改良により変更される場合がある)		
				14	解析又は予報の作成処理識別符		missing			
				15~16	観測資料の参照時刻からの締切時間(時)			2		
				17	観測資料の参照時刻からの締切時間(分)			30		
				18	期間の単位の指示符	符号表4.4	1	時		
				19~22	予報時間			3		
				23	第一固定面の種類	符号表4.5	2			
				24	第一固定面の尺度因子			2		
				25~28	第一固定面の尺度付きの値			2		
				29	第二固定面の種類	符号表4.5	missing			
				30	第二固定面の尺度因子			missing		
				31~34	第二固定面の尺度付きの値			missing		
				35	アンサンブル予報の種類	符号表4.6	4	1=摂動を与えない低分解コントロール、2=負の摂動予報、3=正の摂動予報		
				36	摂動番号			4		
		37	アンサンブルにおける予報の数			51				
		38~39	全時間間隔の終了時(年)			3				
		40	全時間間隔の終了時(月)			3				
		41	全時間間隔の終了時(日)			3				
		42	全時間間隔の終了時(時)			3				
		43	全時間間隔の終了時(分)			3				
		44	全時間間隔の終了時(秒)			3				
		45	統計を算出するために使用した時間間隔を記述する期間の仕様の数			1				
		46~49	統計処理における欠測資料の総数			0				
		50	統計処理の種類			1				
		51	統計処理の時間増分の種類			2				
		52	統計処理の時間の単位の指示符			1				
		53~56	統計処理した期間の長さ			3				
		57	連続的な資料場間の増分に関する時間の単位の指示符			1				
		58~61	連続的な資料場間の時間の増分			0				
		第5節	資料表現節	1~4	節の長さ			21		
				5	節番号			5		
				6~9	全資料点数			10512	144x73	
				10~11	資料表現テンプレート番号	符号表5.0	0	格子点資料・単純圧縮		
				12~15	参照値(R)(IEEE 3.2ビット浮動小数点)			Rは可変		
				16~17	二進尺度因子(E)			Eは可変		
				18~19	十進尺度因子(D)			Dは可変		
				20	単純圧縮による各圧縮値のビット数			12		
				21	原資料場の値の種類	符号表5.1	0	浮動小数点		
				第6節	ビットマップ節	1~4	節の長さ			6
						5	節番号			6
6	ビットマップ指示符							255	ビットマップを適用せず	
第7節	資料節	1~4	節の長さ			15773				
		5	節番号			7				
		6~n	単純圧縮オクテット列			X-	単純圧縮された格子点値の列			
第8節	終端節	1~4	7777		"7777"	国際アルファベットNo.5(CCITT IA5)				

メンバー、要素および水水平面毎に、第4節～第7節を繰り返す

(注) 値が"missing"の場合、そのデータは全ビット1の値、英数字の変数名や"\*\*\*\*\*"は可変を示す。

2.2 週間アンサンブル数値予報モデルの日本域に用いるGRIB2のフォーマットおよびテンプレートの詳細

節番号	節の名称 該当テンプレート	オクテット	内容	表	値	備考			
第0節	指示節	1~4	GRIB		"GRIB"	国際アルファベットNo.5(CCITT IA5)			
		5~6	保留		missing				
		7	資料分野		符号表0.0	0	気象分野		
		8	GRIB版番号			2			
		9~16	GRIB報全体の長さ			172028723			
第1節	識別節	1~4	節の長さ			21			
		5	節番号			1			
		6~7	作成中枢の識別		共通符号表C-1	34	東京		
		8~9	作成副中枢			0			
		10	GRIBマスター表バージョン番号		符号表1.0	2	現行運用バージョン番号		
		11	GRIB地域表バージョン番号		符号表1.1	1	地域表バージョン1		
		12	参照時刻の意味		符号表1.2	1	予報の開始時刻		
		13~14	資料の参照時刻(年)			*****			
		15	資料の参照時刻(月)			*****			
		16	資料の参照時刻(日)			*****			
		17	資料の参照時刻(時)			*****			
		18	資料の参照時刻(分)			*****			
		19	資料の参照時刻(秒)			*****			
		20	作成ステータス		符号表1.3	T	0=現業プロダクト, 1=現業的試験プロダクト		
		21	資料の種類		符号表1.4	5	コントロール及び摂動予報プロダクト		
		第2節	地域使用節	不使用				省略	
		第3節	格子系定義節	1~4	節の長さ			72	
				5	節番号			3	
				6	格子系定義の出典		符号表3.0	0	符号表3.1参照
7~10	資料点数					2920	73x40		
11	格子点数を定義するリストのオクテット数					0			
12	格子点数を定義するリストの説明					0			
13~14	格子系定義テンプレート番号				符号表3.1	0	緯度・経度格子		
15	地球の形状				符号表3.2	6	半径6,371kmの球体と仮定した地球		
16	地球球体の半径の尺度因子					missing			
17~20	地球球体の尺度付き半径					missing			
21	地球回転楕円体の長軸の尺度因子					missing			
22~25	地球回転楕円体の長軸の尺度付きの長さ					missing			
26	地球回転楕円体の短軸の尺度因子					missing			
27~30	地球回転楕円体の短軸の尺度付きの長さ					missing			
31~34	緯線に沿った格子点数					73			
35~38	経線に沿った格子点数					40			
39~42	原作成領域の基本角					0			
43~46	端点の経度及び緯度並びに方向増分の定義に使われる基本角の細分					missing			
47~50	最初の格子点の緯度				10**-6度単位	71250000	北緯71.25度		
51~54	最初の格子点の経度				10**-6度単位	90000000	東経90度		
55	分解能及び成分フラグ				フラグ表3.3	0x30			
56~59	最後の格子点の緯度				10**-6度単位	22500000	北緯22.5度		
60~63	最後の格子点の経度				10**-6度単位	180000000	東経180度		
64~67	方向の増分				10**-6度単位	1250000	1.25度		
68~71	方向の増分				10**-6度単位	1250000	1.25度		
72	走査モード				フラグ表3.4	0x00			
第4節	プロダクト定義節			1~4	節の長さ			*****	
				5	節番号			4	
				6~7	テンプレート直後の座標値の数			0	
				8~9	プロダクト定義テンプレート番号		符号表4.0	*****	1=ある時刻の, ある水平面における個々のアンサンブル予報, 11=連続又は不連続な時間間隔の水平面における個々のアンサンブル
				10	パラメータカテゴリー		符号表4.1	1	
				11	パラメータ番号		符号表4.2	1	
				12	作成処理の種類		符号表4.3	4	アンサンブル予報
				13	背景作成処理識別符		JMA定義	*****	12=週間アンサンブル予報(数値予報モデルの改良により変更される場合がある)
				14	解析又は予報の作成処理識別符			missing	
				15~16	観測資料の参照時刻からの締切時間(時)			2	
				17	観測資料の参照時刻からの締切時間(分)			30	
				18	期間の単位の指示符		符号表4.4	1	時
				19~22	予報時間			3	
				23	第一固定面の種類		符号表4.5	2	
				24	第一固定面の尺度因子			2	
				25~28	第一固定面の尺度付きの値			2	
		29	第二固定面の種類		符号表4.5	missing			
		30	第二固定面の尺度因子			missing			
		31~34	第二固定面の尺度付きの値			missing			
		35	アンサンブル予報の種類		符号表4.6	4	1=摂動を与えない低分解コントロール, 2=負の摂動予報, 3=正の摂動予報		
		36	摂動番号			4			
		37	アンサンブルにおける予報の数			51			
		38~39	全時間間隔の終了時(年)			3			
		40	全時間間隔の終了時(月)			3			
		41	全時間間隔の終了時(日)			3			
		42	全時間間隔の終了時(時)			3			
		43	全時間間隔の終了時(分)			3			
		44	全時間間隔の終了時(秒)			3			
		45	統計を算出するために使用した時間間隔を記述する期間の仕様の数			1			
		46~49	統計処理における欠測資料の総数			0			
		50	統計処理の種類			1			
		51	統計処理の時間増分の種類			2			
		52	統計処理の時間の単位の指示符			1			
		53~56	統計処理した期間の長さ			3			
		57	連続的な資料場間の増分に関する時間の単位の指示符			1			
		58~61	連続的な資料場間の時間の増分			0			
		第5節	資料表現節	1~4	節の長さ			21	
				5	節番号			5	
				6~9	全資料点数			2920	73x40
				10~11	資料表現テンプレート番号		符号表5.0	0	格子点資料・単純圧縮
				12~15	参照値(R)(IEEE 3.2ビット浮動小数点)			R	Rは可変
				16~17	二進尺度因子(E)			E	Eは可変
18~19	十進尺度因子(D)					D	Dは可変		
20	単純圧縮による各圧縮値のビット数					12			
21	原資料場の値の種類				符号表5.1	0	浮動小数点		
第6節	ビットマップ節			1~4	節の長さ			6	
				5	節番号			6	
				6	ビットマップ指示符			255	ビットマップを適用せず
第7節	資料節	1~4	節の長さ			4385			
		5	節番号			7			
		6~n	単純圧縮オクテット列			X	単純圧縮された格子点値の列		
第8節	終端節	1~4	7777		"7777"	国際アルファベットNo.5(CCITT IA5)			

メンバー、要素および水水平面毎に、第4節～第7節を繰り返す

(注) 値が"missing"の場合、そのデータは全ビット1の値、英数字の変数名や"\*\*\*\*\*"は可変を示す。

1 要素の表現 (第4節 10~11オクテットについて)

	10オクテット パラメータカテゴリ (符号表4.1)	11オクテット パラメータ番号 (符号表4.2)
気温	0 (温度)	0 (温度 K)
相対湿度	1 (湿度)	1 (相対湿度 %)
積算降水量	"	8 (総降水量 $\text{kg}\cdot\text{m}^{-2}$ )
風の東西成分	2 (運動量)	2 (風のu成分 $\text{m}/\text{s}$ )
風の南北成分	"	3 (風のv成分 $\text{m}/\text{s}$ )
上昇流	"	8 (鉛直速度(気圧) $\text{Pa}/\text{s}$ )
地上気圧	3 (質量)	0 (気圧 Pa)
海面更正気圧	"	1 (海面更正気圧 Pa)
高度	"	5 (ジオポテンシャル高度 gpm)
全雲量	6 (雲)	1 (全雲量 %)
下層雲量	"	3 (下層雲量 %)
中層雲量	"	4 (中層雲量 %)
上層雲量	"	5 (上層雲量 %)

2 固定面の表現 (第4節 23~28オクテットについて)

	23オクテット 第一固定面の種類 (符号表4.5)	24オクテット 第一固定面の 尺度因子	25~28オクテット 第一固定面の 尺度付きの値
地面	1 (地面又は水面)	missing	missing
平均海面	101 (平均海面)	missing	missing
地上10m (風)	103 (地上からの特定高度面)	0	10
地上2m (気温,RH)	103 (地上からの特定高度面)	0	2
1000 hPa	100 (等圧面 Pa)	-2	1000
975 hPa	"	"	975
950 hPa	"	"	950
925 hPa	"	"	925
900 hPa	"	"	900
850 hPa	"	"	850
800 hPa	"	"	800
700 hPa	"	"	700
600 hPa	"	"	600
500 hPa	"	"	500
400 hPa	"	"	400
300 hPa	"	"	300
250 hPa	"	"	250
200 hPa	"	"	200
150 hPa	"	"	150
100 hPa	"	"	100

### 3 時刻の表現 (特に降水量について)

プロダクト定義節(第4節)は、要素が積算降水量の場合は、テンプレート4.11、その他の要素ではテンプレート4.1を用いる。

テンプレート4.1 の場合、参照時刻(第1節)に予報時間(第4節)を加えた時刻が資料節の内容になる。

テンプレート4.11 即ち降水量の場合、参照時刻(第1節)に予報時間(第4節)を加えた時刻から全期間の終了時(第4節)が示す時刻までの降水量が資料節の内容になる。

アンサンブル数値予報モデルGPVにおいて降水量は初期時刻からの積算降水量の値として表現される。そのためテンプレート4.11の予報時間(19~22オクテット)の値は、全て0である。

(2006年1月10日12UTCを初期値とする降水量の場合)

第1節	オクテット 13~19	参照時刻	2006.01.10.12:00		
第4節	18	期間の単位の 指示符	1	1	1
第4節	19~22	予報時間	0	0	0
第4節	35~41	全時間の終了 統計処理した 期間の長さ	2006.01.10.18:00	2006.01.11.00:00	2006.01.11.06:00
第4節	53~56	統計処理した 期間の長さ	6	12	18

(単位は  
時間)

統計期間	開始時刻 + 終了時刻	2006.01.10.12:00 2006.01.10.18:00	2006.01.10.12:00 2006.01.11.00:00	2006.01.10.12:00 2006.01.11.06:00
	資料節の内容	初期時刻から 6時間後までの 積算降水量	初期時刻から 12時間後までの 積算降水量	初期時刻から 18時間後までの 積算降水量

### 4 メンバーの表現(第4節 35, 36オクテットについて)

全部で51あるメンバーは、第4節の35, 36オクテットで識別する。

第4節	オクテット 35	アンサンブル予報 の種類	1 (コントロール)	2 (負の摂動予報)	3 (正の摂動予報)
第4節	36	摂動番号	0	1~25	1~25

## GRIB2 ファイル サンプルデコード処理プログラム

このプログラムの全部又は一部を利用してもかまいませんが、利用したことによって利用者が被った直接的又は間接的ないかなる損害についても、気象庁は一切責任を負いません。

また、プログラムに関する個別の対応は行いかねますので、ご容赦願います。

```

1  /*-----
2  2005.07.25      sample_decoder.c
3
4  GRIB2 ファイル サンプルデコード処理プログラム
5
6  このプログラムの全部又は一部を利用してもかまいませんが、利用したこ
7  とによって利用者が被った直接的又は間接的ないかなる損害についても、
8  気象庁は一切責任を負いません。
9  また、プログラムに関する個別の対応は行いかねますので、ご容赦願います。
10
11  利用方法
12
13  ANSI 準拠の C コンパイラでコンパイルして下さい。
14  GRIB2 ファイルのファイル名を引数に与えて実行すると、
15  ファイルの内容が表示されます。
16  -----*/
17
18  # include <stdio.h>
19  # include <stdlib.h>
20  # include <math.h>
21
22
23  /* 1,2,4,8 : signed integer  (x byte)
24             u : unsigned integer (1 byte)
25             S : unsigned integer (2 byte)
26             C : character      (4 byte)
27             R : float          (4 byte) */
28
29  char *sectionFormat[9] = {
30  "C2uu8",          /* Section 0 */
31  "4u2uuu2uuuuuu", /* Section 1 */
32  "",              /* Section 2 */
33  "4uu4uuS",       /* + Section 3 */
34  "4u2S",          /* + Section 4 */
35  "4u4S",          /* + Section 5 */
36  "4uu",           /* + Section 6 */
37  "4u",            /* + Section 7 */
38  "C"              /* Section 8 */
39  };
40
41  typedef struct { int secno, templat; char *format; } TemplateFormat;
42
43  TemplateFormat templateFormat[] = {
44  { 3, 0, "uu4u4u444444u4444u" },
45  { 4, 0, "uuuuu2uu4u14u14" },
46  { 4, 1, "uuuuu2uu4u14u14uuu" },
47  { 4, 8, "uuuuu2uu4u14u142uuuuuu4uuu4u4" },
48  { 4,11, "uuuuu2uu4u14u14uuu2uuuuuu4uuu4u4" },
49  { 5, 0, "R22uu" },
50  { 0, 0, "" }
51  };
52
53  static int isLittleEndian;
54
55  # define FIT_BYTE_ORDER(pointer,size) if(isLittleEndian)swabN(pointer,size,1)
56
57  /* reverse byte order */
58  static void
59  swabN( void *buf, int size, int nn )
60  {
61  char *ba, *bb, *buf2 = buf;

```

```

62     while( nn-- ) {
63         bb = ( ba = buf2 ) + size -1;
64         do {
65             char a;
66             a = *ba;
67             *ba = *bb;
68             *bb = a;
69         } while( ++ba < --bb );
70         buf2 += size;
71     }
72 }
73
74 int
75 read_section_0( FILE *fp, void **sec_buffer )
76 {
77     char *bufr;
78     *sec_buffer = realloc( *sec_buffer, 16 );
79     bufr = *sec_buffer;
80     if( fread( bufr, 1, 16, fp ) != 16 ||
81         strcmp( bufr, "GRIB", 4 ) != 0 ) {
82         fprintf( stderr, "Really GRIB file ?%n" );
83         exit(1);
84     }
85     return 0;
86 }
87
88 int
89 read_section_X( FILE *fp, void **sec_buffer )
90 {
91     int length;
92     unsigned char *bufp;
93
94     fread( &length, 4, 1, fp );
95     if( strcmp( (char *)&length, "7777", 4 ) == 0 ) {
96         *sec_buffer = realloc( *sec_buffer, 4 );
97         strncpy( *sec_buffer, "7777", 4 );
98         return 8;
99     }
100     FIT_BYTE_ORDER( &length, 4 );
101
102     *sec_buffer = realloc( *sec_buffer, length );
103     bufp = *sec_buffer;
104     if( fread( bufp + 4, 1, length - 4, fp ) != length - 4 ) {
105         fprintf( stderr, "Unexpected EOF%n" );
106         exit(1);
107     }
108     FIT_BYTE_ORDER( &length, 4 );
109     memcpy( bufp, &length, 4 );
110
111     return (int)bufp[4]; /* section No. */
112 }
113
114 void
115 decode_buf( char *sec_buffer, int *index, char *format, double **dvpp )
116 {
117     unsigned char buffer[8];
118     int size, ii, jj, missing;
119     double *dvp, dd;
120
121     dvp = *dvpp;
122

```

```

123     for( ii = *index ; *format ; format++ ) {
124         switch( *format ) {
125             case '1' :
126                 case '2' :
127                 case '4' :
128                 case '8' : size = *format - '0'; break;
129                 case 'u' : size = 1; break;
130                 case 'S' : size = 2; break;
131                 case 'R' :
132                 case 'C' : size = 4; break;
133                 default :
134                     fprintf( stderr, "Internal Error !%n" );
135                     exit(9);
136             }
137         memcpy( buffer, sec_buffer+ii, size );
138
139         missing = 1;
140         for( jj = 0 ; jj < size ; jj++ )
141             missing &= ( buffer[jj] == 0xFF );
142
143         switch( *format ) {
144             case 'u' : dd = *buffer;                break;
145             case 'S' : dd = (buffer[0]<<8) + buffer[1]; break;
146
147             case '1' :
148             case '2' :
149             case '4' :
150             case '8' : dd = buffer[0] & 0x7F;
151                 for( jj = 1 ; jj < size ; jj++ )
152                     dd = dd * 256.0 + buffer[jj];
153                 if( *buffer & 0x80 ) dd = -dd;    break;
154
155             case 'R' : FIT_BYTE_ORDER( buffer, size );
156                 dd = *(float *)buffer;        break;
157
158             case 'C' : memcpy( &dd, buffer, size );    break;
159         }
160         if( size == 1 ) printf( "   %4d   : ", ii+1 );
161         else             printf( "%4d .. %-4d: ", ii+1, ii+size );
162
163             if( missing )     printf( "missing %n" );
164             else if( *format == 'R' ) printf( "%f%n", dd );
165             else if( *format == 'C' ) printf( "%.4s'%n", &dd );
166             else             printf( "%.0f%n", dd );
167
168             *dvp++ = dd;
169             ii += size;
170         }
171         *index = ii;
172         *dvpp = dvp;
173     }
174
175 void
176 decode_section( int secno, char *sec_buffer, double **double_values )
177 {
178     int     index, length;
179     double *dvp;
180     char   *format;
181     TemplateFormat *tftp;
182
183     switch( secno ) {

```

```

184     case 0 : length = 5; break;
185     case 1 : length = 15; break;
186     case 7 : length = 2; break;
187     case 8 : length = 1; break;
188     default :
189         memcpy( &length, sec_buffer, 4 );
190         FIT_BYTE_ORDER( &length, 4 );
191     }
192     dvp = realloc( *double_values, sizeof(double) * length );
193     *double_values = dvp;
194
195     printf( "== SECTION %d == %n", secno );
196
197     format = sectionFormat[secno];
198     index = 0;
199
200     decode_buf( sec_buffer, &index, format, &dvp );
201
202     if( 3 <= secno && secno <= 5 ) {
203         int templat = (int)dvp[-1];
204         for( tfp = templateFormat ; tfp->secno ; tfp++ ) {
205             if( tfp->secno == secno &&
206                 tfp->templat == templat ) break;
207         }
208         if( ! tfp->secno ) {
209             fprintf( stderr, "No Information about Template %d.%d%#n",
210                     secno, templat );
211             return;
212         }
213         decode_buf( sec_buffer, &index, tfp->format, &dvp );
214     }
215 }
216
217 void
218 unpack_data( const char *sec_7, const double *values_5, float **out )
219 {
220     double rr, ee, dd, pow_2_e, pow_10_d;
221     float *op;
222     int num, nbit, ii;
223     unsigned int uw, mask;
224     const unsigned char *uc;
225
226     uc = (const unsigned char *)sec_7 + 5;
227
228     num = values_5[2];
229     rr = values_5[4];
230     ee = values_5[5];
231     dd = values_5[6];
232     nbit = values_5[7];
233     pow_2_e = pow( 2.0, ee);
234     pow_10_d = pow(10.0, dd);
235
236     *out = realloc( *out, sizeof(**out) * num );
237     op = *out;
238
239     mask = 1;
240     for( ii = 0 ; ii < nbit - 1 ; ii++ )
241         mask = mask << 1 | 1;
242
243     for( ii = 0 ; ii < num ; ii++ ) {
244         memcpy( &uw, &uc[(nbit*ii)/8], 4 );

```

```

245     FIT_BYTE_ORDER( &uw, 4 );
246     uw = ( uw>>(32-nbit-(nbit*ii)%8) ) & mask;
247     *op++ = ( rr + pow_2_e * uw ) / pow_10_d;
248 }
249 }
250
251 # include <float.h>
252
253 char *
254 elem_name( const double *sec4 )
255 {
256     static char name_buf[256];
257     char level[256], elem[256], *elem;
258     int cc, nn, tt;
259
260     typedef struct { int category, number; char *name; } ETable;
261
262     static ETable etable [] = {
263         { 0, 0, "Temperature" },
264         { 1, 1, "RelativeHumidity" },
265         { 1, 8, "TotalPrecipitation" },
266         { 2, 2, "UofWind" },
267         { 2, 3, "VofWind" },
268         { 2, 8, "VerticalVelocity(Pressure)" },
269         { 3, 0, "Pressure" },
270         { 3, 1, "PressureReducedToMSL" },
271         { 3, 5, "GeopotentialHeight" },
272         { 6, 1, "TotalCloudCover" },
273         { 6, 3, "LowCloudCover" },
274         { 6, 4, "MediumCloudCover" },
275         { 6, 5, "HighCloudCover" },
276         { 0, 0, NULL }
277     };
278     ETable *etp;
279
280     cc = sec4[4];
281     nn = sec4[5];
282     for( etp = etable ; etp->name ; etp++ )
283         if( etp->category == cc && etp->number == nn ) {
284             elem = etp->name;
285             break;
286         }
287     if( ! etp->name ) {
288         fprintf( stderr, "No information about Product category %d number %d\n",
289             cc, nn );
290         sprintf( elem, "Cat(%d)num(%d)", cc, nn );
291         elem = elem;
292     }
293     if( sec4[13] == 1 ) strcpy( level, "Surface" );
294     else if( sec4[13] == 101 ) strcpy( level, "MeanSeaLevel" );
295     else if( sec4[13] == 103 )
296         sprintf( level, "%.1fm", sec4[15] / pow(10., sec4[14]) );
297     else if( sec4[13] == 100 )
298         sprintf( level, "%.0fhPa", sec4[15] / pow(10., sec4[14])/100. );
299     else
300         sprintf( level, "UnknownFixedsurface(%.0f)", sec4[13] );
301
302     switch( (int)sec4[3] ) {
303     case 8 : tt = sec4[12] + sec4[30]; break;
304     case 11 : tt = sec4[12] + sec4[33]; break;
305     default : tt = sec4[12]; break;
306     }

```

```

306     switch( (int)sec4[3] ) {
307     char *ensm;
308     case 1: case 11:
309         switch( (int)sec4[19] ) {
310             case 0 : ensm = "H"; break;
311             case 1 : ensm = "" ; break;
312             case 2 : ensm = "m"; break;
313             case 3 : ensm = "p"; break;
314         }
315         sprintf( name_buf, "%s_%s_T%d_M%.0f%s",
316                 elem, level, tt, sec4[20], ensm ); break;
317     default:
318         sprintf( name_buf, "%s_%s_T%d", elem, level, tt ); break;
319     }
320     return name_buf;
321 }
322
323 void
324 show_data_statistics( const double *values_4, const float *unpacked_data, int num )
325 {
326     int ii;
327     double sum = 0, min = DBL_MAX, max = -DBL_MAX;
328
329     printf( " << %s >> %n", elem_name( values_4 ) );
330
331     for( ii = 0 ; ii < num ; ii++ ) {
332         double ff = unpacked_data[ii];
333         sum += ff;
334         if( min > ff ) min = ff;
335         if( max < ff ) max = ff;
336     }
337     printf( " ( num = %d, min = %f, max = %f, average = %f )%n",
338            num, min, max, sum / num );
339 }
340
341 void
342 save_float_file( const double *values_4, const float *unpacked_data, int num )
343 {
344     FILE *fpout;
345     char  fileName[256];
346
347     sprintf( fileName, "%s.dat", elem_name( values_4 ) );
348
349     fprintf( stderr, "output '%s'%n", fileName );
350
351     if( ( fpout = fopen( fileName, "wb" ) ) == NULL ) {
352         fprintf( stderr, "file '%s' open error! %n", fileName );
353         exit(1);
354     }
355     fwrite( unpacked_data, sizeof(float), num, fpout );
356     fclose( fpout );
357 }
358
359 int main( int argc, char *argv[] )
360 {
361     FILE *fpin;
362     char *fileName;
363     int  secno, ii;
364     void *sec_buffer = NULL;
365     float *unpacked_data = NULL;
366     double *sec_double_value[9];

```

```

367
368     /* check system byte order */
369     isLittleEndian = 1;
370     isLittleEndian = *(char *)&isLittleEndian;
371
372     for( ii = 0 ; ii < sizeof(sec_double_value)/sizeof(*sec_double_value) ; )
373         sec_double_value[ii++] = NULL;
374     if( argc == 1 ) {
375         fprintf( stderr, "¥n¥n usage: %s 'grib2 file name'¥n¥n", argv[0] );
376         exit(1);
377     }
378     fileName = argv[1];
379     if( ( fpin = fopen( fileName, "rb" ) ) == NULL ) {
380         fprintf( stderr, "grib2 file '%s' open error! ¥n", fileName );
381         exit(1);
382     }
383     secno = read_section_0( fpin, &sec_buffer );
384     decode_section( secno, sec_buffer, &sec_double_value[secno] );
385
386     while( secno = read_section_X( fpin, &sec_buffer ) ) {
387         decode_section( secno, sec_buffer, &sec_double_value[secno] );
388         if( secno == 8 ) break;
389         if( secno == 7 ) {
390             unpack_data( sec_buffer, sec_double_value[5], &unpacked_data );
391
392             show_data_statistics( sec_double_value[4], unpacked_data,
393                                 (int)sec_double_value[5][2] );
394             /*
395             save_float_file ( sec_double_value[4], unpacked_data,
396                             (int)sec_double_value[5][2] );
397             */
398         }
399     }
400     fclose(fpin);
401     return 0;
402 }

```

平成 17 年 9 月  
気象庁地球環境・海洋部

## 1 か月予報アンサンプル格子点値および 1 か月予報ガイダンスの変更について

### 1. 変更概要

#### 1 か月予報アンサンプル格子点値

配信要素及びメンバー数を増やすとともに、ファイル形式についても、利用しやすいよう GRIB1、2 による提供を行います。変更点は以下のとおりです。配信ファイル形式の詳細については、添付資料 4 を参照してください。

#### 1 か月予報メンバー別全球格子点値（現・メンバー別格子点値）

- ・配信要素数の増加（16 要素 36 要素）
- ・メンバー数の増加（26 メンバー 50 メンバー）
- ・ファイルフォーマットの変更（気象庁の独自形式 GRIB2）
- ・ファイル名の変更（気象庁ファイル命名規則に従ったファイル名に変更）
- ・データ量（220MB 1GB）

#### 1 か月予報アンサンプル統計格子点値（現・アンサンプル格子点値）

##### 【全球または北半球データ】

- ・ファイルフォーマットの変更（気象庁の独自形式 GRIB1）
- ・ファイル名の変更（気象庁ファイル命名規則に従ったファイル名に変更）
- ・データ量はほぼ同じ（1MB）

##### 【極東域データ】

18 年 3 月以降クラスター分析処理を中止するため廃止します。アンサンプルメンバー別の情報が必要な場合は「1 か月予報メンバー別全球格子点値」をご利用ください。

および の名称は、他の予報格子点値などとの混乱を避けるため現在の（ ）内の名称から変更します。

#### 1 か月予報ガイダンス

メンバー数の増加、予報時間間隔の細分化を行うとともに、ファイル形式についても、利用しやすいよう csv 形式による提供を行います。変更点は以下のとおりです。配信ファイル形式の詳細については添付資料 5 を参照してください。

- ・メンバー数の増加（26 メンバー 50 メンバー）

- ・予報の時間間隔の細分化(データにより1日間、1週間、2週間間隔 全て1日間間隔)
- ・ファイルフォーマットの変更(気象庁の独自形式 CSV(テキスト)形式)
- ・ファイル名の変更(気象庁ファイル命名規則に従ったファイル名に変更)
- ・データ量(1.6MB 20MB)

## 2. 提供開始日

平成18年3月3日(金)から配信を予定しています。

なお、提供開始に先立ち試験配信を行う予定です。試験配信開始日については別途お知らせします。

## 3. サンプルデータ

サンプルデータを媒体(CD-ROM 2枚)に収録して気象業務支援センターから提供しますので、必要な場合はご利用下さい。収録内容の詳細については添付資料8を参照してください。

## 4. 解読処理

1か月予報アンサンブル格子点値は解読(デコード)処理が必要です。データの解読処理については、添付資料6および7をそれぞれ参照してください。また、参考までに解読処理用サンプルプログラムのWindows実行形式を上記のサンプルデータとともにCD-ROMに収録しましたので、ご利用ください。

## 1 か月予報アンサンブル格子点値の解説

## 1. 概要

1 か月予報アンサンブル格子点値には、1 か月予報メンバー別全球格子点値と1 か月予報アンサンブル統計格子点値がある。

## (1) 1 か月予報メンバー別全球格子点値 (日別)

作成回数 : 週1回  
 予報時間 : 34日間 (1日間間隔)  
 アンサンブルメンバー数 : 50メンバー (水曜日 25メンバー、木曜日 25メンバー - 合計 50メンバー)  
 格子系 : 等緯度経度 (2.5度格子)  
 領域 : 全球  
 データ内容 :  
 地上要素

	海面更正気圧*	積算降水量	2 m気温
地上			

## 気圧面要素

気圧面 (hPa)	高度*	風	気温*	相対湿度
1000				
850				
700				
500				
300				
200				
100				

東西方向と南北方向の2要素

\*海面更正気圧、高度、気温 (2 m気温を除く) は系統誤差補正済み。

## (2) 1 か月予報アンサンブル統計格子点値

作成回数 : 週1回  
 予報期間 : 4週間 (1週間間隔または2週間間隔)  
 統計処理 :  
 (メンバー) アンサンブル平均  
 (期間) 1週間平均、2週間平均または4週間平均  
 格子系 : 等緯度経度 (2.5度格子)  
 領域 : 全球  
 データ内容 :  
 地上要素

	海面更正気圧	海面更正気圧平年偏差	積算降水量
地上			

## 気圧面要素

気圧面 (hPa)	高度	高度平年偏差	風	気温	気温平年偏差	相対湿度	その他
850							気温スプレッド*1
500							高度スプレッド 高度高偏差確率*2
200							
100							

は 2 要素分のデータ（風の場合，東西方向と南北方向の 2 要素）

統計処理、予報の時間間隔は、要素により異なっている、詳細な内容は以下のとおり。  
（全球または北半球域）

要素		レベル (hPa)	領域	予報対象期間
アンソブル平均値 (7日平均値場)	海面更正気圧、 積算降水量	-	全球 2.5x2.5 度	予報初日から 0-6, 7-13, 14-20, 21-27 日
	気温、相対湿度、 風(東西成分、南北成分)	850		
	シ 林° テンシャル高度	500,100		
	風(東西成分、南北成分)	200		
	海面更正気圧の平年偏差	-	北半球 2.5x2.5 度	
	気温の平年偏差	850		
	シ 林° テンシャル高度の平年偏差	500,100		
アンソブルメンバー間の スプレッド	海面更正気圧	-	北半球 2.5x2.5 度	予報初日から 0-6, 7-13, 14-20, 21-27, 0-13, 14-27, 0-27 日
	気温	850		
	シ 林° テンシャル高度	500		
高偏差確率		500		

- \* 1 スプレッド : 予報メンバーの標準偏差を自然変動の標準偏差で規格化した値。
- \* 2 高偏差確率 : 予想される偏差の絶対値が自然変動の標準偏差の 0.5 倍を上回る確率。

## 2. ファイルフォーマット等の詳細

## (1) 1か月予報アンサンブル統計格子点値

- ファイル名 : 「1か月予報アンサンブル格子点値ファイル名」参照  
 レコード形式 : 「国際気象通報式 FM92 GRIB 二進形式格子点資料気象通報式(第1版)」  
 GBIB(予報日時(期間)別, 領域別, 層別, 物理量別に格納)  
 (GRIBのデータ表現についての補足説明を別紙1に示す)  
 ファイルサイズ : 1MB(124GRIB)

## (2) 1か月予報メンバー別全球格子点値

- ファイル名 : 「1か月予報アンサンブル格子点値ファイル名」参照  
 レコード形式 : 「国際気象通報式 FM92 GRIB 二進形式格子点資料気象通報式(第2版)」  
 (GRIB2)  
 「1か月予報メンバー別全球格子点値ファイルにおけるGRIB2第4節の補足説明(別紙2)」参照  
 ファイルサイズ : 1ファイルあたり28.5MB、36ファイルの合計約1GB

## 1か月予報アンサンブル格子点値ファイル名

	ファイル名称	サイズ (MB)	データ内容
1 か 月 予 報 メ ン バ ー 別 全 球 格 子 点 値	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lsurf_P pp_Emb_grib2.bin	28.5	海面更正気圧
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lsurf_P rr_Emb_grib2.bin	28.5	積算降水量
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lh2_Ptt _Emb_grib2.bin	28.5	2m 気温
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp1000_ Phh_Emb_grib2.bin	28.5	1000hPa 高度
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp1000_ Pwu_Emb_grib2.bin	28.5	1000hPa 風 (東向き成分)
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp1000_ Pwv_Emb_grib2.bin	28.5	1000hPa 風 (北向き成分)
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp1000_ Ptt_Emb_grib2.bin	28.5	1000hPa 気温
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp1000_ Prh_Emb_grib2.bin	28.5	1000hPa 相対湿度
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp850_P hh_Emb_grib2.bin	28.5	850hPa 高度
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp850_P wu_Emb_grib2.bin	28.5	850hPa 風 (東向き成分)
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp850_P wv_Emb_grib2.bin	28.5	850hPa 風 (北向き成分)
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp850_P tt_Emb_grib2.bin	28.5	850hPa 気温
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp850_P rh_Emb_grib2.bin	28.5	850hPa 相対湿度
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp700_P hh_Emb_grib2.bin	28.5	700hPa 高度
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp700_P wu_Emb_grib2.bin	28.5	700hPa 風 (東向き成分)
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp700_P wv_Emb_grib2.bin	28.5	700hPa 風 (北向き成分)

	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp700_P tt_Emb_grib2.bin	28.5	700hPa 気温
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp700_P rh_Emb_grib2.bin	28.5	700hPa 相対湿度
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp500_P hh_Emb_grib2.bin	28.5	500hPa 高度
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp500_P wu_Emb_grib2.bin	28.5	500hPa 風 (東向き成分)
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp500_P wv_Emb_grib2.bin	28.5	500hPa 風 (北向き成分)
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp500_P tt_Emb_grib2.bin	28.5	500hPa 気温
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp500_P rh_Emb_grib2.bin	28.5	500hPa 相対湿度
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp300_P hh_Emb_grib2.bin	28.5	300hPa 高度
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp300_P wu_Emb_grib2.bin	28.5	300hPa 風 (東向き成分)
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp300_P wv_Emb_grib2.bin	28.5	300hPa 風 (北向き成分)
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp300_P tt_Emb_grib2.bin	28.5	300hPa 気温
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp300_P rh_Emb_grib2.bin	28.5	300hPa 相対湿度
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp200_P hh_Emb_grib2.bin	28.5	200hPa 高度
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp200_P wu_Emb_grib2.bin	28.5	200hPa 風 (東向き成分)
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp200_P wv_Emb_grib2.bin	28.5	200hPa 風 (北向き成分)
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp200_P tt_Emb_grib2.bin	28.5	200hPa 気温
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp100_P hh_Emb_grib2.bin	28.5	100hPa 高度
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp100_P wu_Emb_grib2.bin	28.5	100hPa 風 (東向き成分)
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp100_P wv_Emb_grib2.bin	28.5	100hPa 風 (北向き成分)
	Z_C_RJTD_yyyyMMddhhmmss_EPS1_MGPV_Rgl_Lp100_P tt_Emb_grib2.bin	28.5	100hPa 気温
1 か 月予 報ア ンサ ンブ ル統 計格 子点 値	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GPV_Rgl_Eem_grib .bin	1	アンサンプル統計値 (GRIB)

全ファイル数 = 36+1 ファイル

( 1 か月予報メンバー別全球、36 要素 ( 36 ファイル ) + 1 か月予報アンサンプル統計 ( 1 ファイル ) )

このファイル名は、国際的な資料交換に用いるため、世界気象機関（WMO）により採用されたファイル命名規則に準拠し、任意部分を当庁において定義したものである（技術情報第 130 号）。

Z\_C : Z と C の間には、アンダースコア “\_” が 2 つ続く “\_\_”  
yyyyMMddhhmmss : 数値予報の初期値年月日時を表す。mmss は 0000 とする。

---

### 用語説明

- ・アンサンブル予報 : 観測（解析）誤差程度のわずかな違いのある複数の初期値をもとに数値予報を行ない、それぞれの結果を統計的に処理する予測手法。
- ・メンバー : アンサンブル予報を構成している個々の予報。
- ・アンサンブル平均 : 各メンバーを平均して求めた予測結果。
- ・スプレッド : 予報メンバーの標準偏差を自然変動の標準偏差で規格化した値。アンサンブル予報を構成しているメンバー間のばらつきの大きさを示す指標。
- ・高偏差確率 : 予想される偏差の絶対値が自然変動の標準偏差の 0.5 倍を上回る確率。

WMO国際気象通報式 F M 9 2  
G R I B - 二進形式格子点資料気象通報式に関する補足説明

1 パラメータの指示符に用いる数字符号

アンサンブル統計格子点値 1 (GRIB)における、第 1 節 (プロダクト定義節) の第 9 オクテット (パラメータの指示符) に用いる第 2 表について、国際標準のパラメータに加え以下のパラメータが追加されている。

数字符号	パラメータ
1 4 0	ジオポテンシャル高度の高偏差確率
1 4 1	気圧のスプレッド
1 4 2	ジオポテンシャル高度のスプレッド
1 4 3	気温のスプレッド

2 期間などを表すパラメータ

1 か月アンサンブル予報では、2 つの初期値時刻からのアンサンブル数値予報を 1 セットとしてプロダクトを作成する。そして、期間平均をとる際は予報対象日をそろえて処理している。

このアンサンブル予報資料を表現するにあたって、アンサンブル統計格子点値 1 (GRIB)においては、第 1 節の 1 3 ~ 2 1 オクテットのパラメータ (参照時刻、平均期間など) は以下のようにセットされている。

オクテット番号	
1 3 ~ 1 7	参照時刻は 2 日目の初期時刻としている。
1 8	期間の単位の指示符は ' 2 ' (1 日単位、第 4 表参照)
1 9 , 2 0	2 つの期間パラメータは P1 (期間平均の最初)、P2 (期間平均の終わり) とし、例えば 7 日間の期間平均の場合は、以下のようにセットする。

P1	P2
2	8
9	1 5
1 6	2 2
2 3	2 9

2 1	期間の指示符については、 降水量については、' 4 ' (積算) それ以外の物理量については、' 3 ' (平均)
-----	---

## 1 か月予報メンバー別全球格子点値ファイルにおけるGRIB2第4節の補足説明

## 1 . GRIB2のフォーマット及びテンプレートの詳細

	節番号	節の名称・ 該当テンプレート	オクテット	内容	表	値	
第4節	プロダクト定義節	1~4	節の長さ		****		
		5	節番号			4	
		6~7	テンプレート直後の座標値の数				0
		8~9	プロダクト定義テンプレート番号	符号表4.0		1	ある時刻の水平面または水平層における個々のアンサンブル予報、コントロール予報及び摂動予報
		10	パラメータカテゴリー	符号表4.1		*	
		11	パラメータ番号	符号表4.2		*	
		12	作成処理の種類	符号表4.3		4	アンサンブル予報
		13	背景作成処理識別符	<b>JMA定義</b>		***	アンサンブル1か月予報
		14	予報の作成処理識別符			missing	
		15~16	観測資料の参照時刻からの締切時間(時)				2
		17	観測資料の参照時刻からの締切時間(分)				30
		18	期間の単位の指示符	符号表4.4		2	日
		19~22	予報時間			****	予報時間
		23	第一固定面の種類	符号表4.5		*	
		24	第一固定面の尺度因子			*	
		25~28	第一固定面の尺度付きの値			*	
		29	第二固定面の種類	符号表4.5		missing	
		30	第二固定面の尺度因子			missing	
		31~34	第二固定面の尺度付きの値			missing	
		35	アンサンブル予報の種類	符号表4.6		*	
36	摂動番号			*			
37	アンサンブルにおける予報の数			*			

## 1 か月予報ガイダンスの解説

### 1. 概要

- 予報初日 : 土曜日  
予報期間 : 31 日 ( 1 日間間隔 )  
統計処理 : 7 日、14 日、28 日の各日別移動平均  
メンバー数 : 50 メンバ  
領域 : 気象庁が 1 か月予報を行なう場合の地方予報区およびその細分地域の 34 地域  
( 北日本、北日本日本海側、北日本太平洋側、東日本、東日本日本海側、東日本太平洋側、西日本、西日本日本海側、西日本太平洋側、南西諸島、北海道地方、北海道日本海側、北海道オホーツク海側、北海道太平洋側、東北地方、東北日本海側、東北太平洋側、東北北部、東北南部、関東甲信地方、北陸地方、東海地方、近畿地方、近畿日本海側、近畿太平洋側、中国地方、山陰、山陽、四国地方、九州北部地方、九州南部地方、九州南部、奄美地方、沖縄地方 )  
要素 : 気温、降水量、日照時間、降雪量<sup>\*1</sup>、晴れ日数<sup>\*2</sup>、降水日数<sup>\*3</sup>、雨日数<sup>\*4</sup>  
気温、降水量、日照時間、降雪量の「高い(多い)」または「低い(少ない)」の階級が出現する確率値

ファイルサイズ : 1 地域あたり 586KB。全 34 地域結合したファイルは約 2MB

なお、各ガイダンスはメンバー別に作成。

- \* 1 降雪量 : 12 月～翌 2 月のみ算出。  
\* 2 晴れ日数 : 日照時間の可照時間に対する割合が 40% 以上の日数。  
\* 3 降水日数 : 日降水量が 1 mm 以上の日数。  
\* 4 雨日数 : 日降水量が 10mm 以上の日数。

## 2. フォーマット等の詳細

## (1) ファイル名

ファイル名は下表のとおりで、予報区の細分地域毎にファイルを作成し、作成したすべてのファイルを圧縮する。

内容	ファイル名
北日本	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJknh00_JRlong_P-all_tablr.csv
北日本日本海側	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJknh01_JRlong_P-all_tablr.csv
北日本太平洋側	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJknh02_JRlong_P-all_tablr.csv
東日本	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJhnh00_JRlong_P-all_tablr.csv
東日本日本海側	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJhnh01_JRlong_P-all_tablr.csv
東日本太平洋側	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJhnh02_JRlong_P-all_tablr.csv
西日本	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJnnh00_JRlong_P-all_tablr.csv
西日本日本海側	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJnnh01_JRlong_P-all_tablr.csv
西日本太平洋側	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJnnh02_JRlong_P-all_tablr.csv
南西諸島	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJnss00_JRlong_P-all_tablr.csv
北海道地方	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJhkd00_JRlong_P-all_tablr.csv
北海道日本海側	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJhkd01_JRlong_P-all_tablr.csv
北海道オホーツク海側	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJhkd03_JRlong_P-all_tablr.csv
北海道太平洋側	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJhkd02_JRlong_P-all_tablr.csv
東北地方	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJthk00_JRlong_P-all_tablr.csv
東北地方日本海側	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJthk01_JRlong_P-all_tablr.csv
東北地方太平洋側	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJthk02_JRlong_P-all_tablr.csv
東北北部	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJthk05_JRlong_P-all_tablr.csv
東南北部	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJthk04_JRlong_P-all_tablr.csv
関東甲信地方	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJtk00_JRlong_P-all_tablr.csv
北陸地方	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJhkr00_JRlong_P-all_tablr.csv
東海地方	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJtki00_JRlong_P-all_tablr.csv
近畿地方	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJknk00_JRlong_P-all_tablr.csv
近畿日本海側	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJknk01_JRlong_P-all_tablr.csv
近畿太平洋側	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJknk02_JRlong_P-all_tablr.csv
中国地方	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJcgg00_JRlong_P-all_tablr.csv
山陰	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJcgg06_JRlong_P-all_tablr.csv
山陽	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJcgg07_JRlong_P-all_tablr.csv
四国地方	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJskk00_JRlong_P-all_tablr.csv
九州北部地方	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJkyh00_JRlong_P-all_tablr.csv
九州南部地方	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJkyn00_JRlong_P-all_tablr.csv
九州南部	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJkyn08_JRlong_P-all_tablr.csv
奄美地方	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJkyn09_JRlong_P-all_tablr.csv
沖縄地方	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_RJokn00_JRlong_P-all_tablr.csv
以上を結合したファイル	Z_C_RJTD_yyyyMMddhhmmss_EPS1_GUID_Rjp_JRlong_P-all_tablr.tar

このファイル名は、国際的な資料交換に用いるため、世界気象機関（WMO）により採用されたファイル命名規則に準拠し、任意部分を当庁において定義したものである（技術情報第130号）

Z\_C : ZとCの間には、アンダースコア“\_”が2つ続く“\_\_”  
 yyyyMMddhhmmss : 1か月予報ガイドスでは、数値予報の初期値年月日時を表す。mmssは0000とする。

## (2) フォーマット

フォーマットは CSV 形式 (カンマで区切られたテキストデータ) である。1 行の構成と行の並び方を以下に示す。

## 行の構成

各ファイルは「初期時刻行」、「予測資料行」の 2 種の行により構成されている。各行の説明を下記に示す。表の 1 段目はカラムの説明、2 段目は文字数、3 段目は数値予報ガイダンスのデータの例を示す。各カラムにデータを右詰で収録し、余りはスペースで埋める。各表の下にカラムの詳細を示した。各表中の「C」はカンマ(,)のカラムを表す。

## 初期時刻行

初期値年	C	初期値月	C	初期値日	C
5	1	5	1	5	1
2005	,	9	,	1	,

初期値年・月・日・時・分は、数値予報ガイダンスでは数値予報モデルの初期値時刻を世界標準時を用いて表す。

## 予測資料行

予測対象期間開始年	C	予測対象期間開始月	C	予測対象期間開始日	C	予測対象期間終了年	C	予測対象期間終了月	C	予測対象期間終了日	C	予測対象期間長	C
5	1	5	1	5	1	5	1	5	1	5	1	5	1
2005	,	8	,	26	,	2005	,	9	,	1	,	7	,

続く

地域番号	C	要素番号*	C	予測式の種別*	C	予測値* (アンサンブル平均)	C	予測値* (メンバ1)	C	予測値* (メンバ2)	C	...	C	予測値* (メンバ50)	C
5	1	5	1	5	1	5	1	5	1	5	1	...	1	5	1
1	,	1	,	1	,	-7	,	-4	,	-10	,	...	,	-6	,

メンバ：数値予報ガイダンスでは、本行の予測値がアンサンブル数値予報のどのメンバに基づくのかをカラムの順で表す。

予測対象期間長：本行が収録している予測結果の対象期間の長さを日を単位として表す。

地域番号：表 1 にまとめた。

要素番号、予測値：予測要素に割り振った番号と予測値の単位について表 2 にまとめた。

予測式の種別：表 3 にまとめた。

\*がついているカラムは値が無い場合"-9999"とする。

## 行の並び

各ファイル中のデータ行の並びは次のとおり。

## 1 か月予報ガイダンス

第 1 行：「初期時刻行」

第 2 行以降は、「予測資料行」を次のとおり並べる。

：日別予測式と期間平均予測式について、要素番号 1~8 ごとに予測対象期間長 7 日、14 日、28 日の順に並べる。

：続いて確率ガイダンスについて、要素番号 11~18 ごとに と同様に並べる。

表1 地域番号と地域名

番号	地域名	番号	地域名	番号	地域名
1	北日本	11	北海道地方	23	近畿地方
2	北日本日本海側	12	北海道日本海側	24	近畿日本海側
3	北日本太平洋側	13	北海道オホーツク海側	25	近畿太平洋側
4	東日本	14	北海道太平洋側	26	中国地方
5	東日本日本海側	15	東北地方	27	山陰
6	東日本太平洋側	16	東北日本海側	28	山陽
7	西日本	17	東北太平洋側	29	四国地方
8	西日本日本海側	18	東北北部	30	九州北部地方
9	西日本太平洋側	19	東北南部	31	九州南部地方
10	南西諸島	20	関東甲信地方	32	九州南部
		21	北陸地方	33	奄美地方
		22	東海地方	34	沖縄地方

表2 予測要素と単位

要素番号	要素	単位
1	気温平年差	0.1
2	降水量平年比	%
3	日照時間平年比	%
4	晴れ日数*1	0.1 日
5	降水日数*2	0.1 日
6	雨日数*3	0.1 日
7	降雪量平年比	%
8	未使用*4	
11	気温「高い」階級の出現する確率	%
12	気温「低い」階級の出現する確率	%
13	降水量「多い」階級の出現する確率	%
14	降水量「少ない」階級の出現する確率	%
15	日照時間「多い」階級の出現する確率	%
16	日照時間「少ない」階級の出現する確率	%
17	降雪量「多い」階級の出現する確率	%
18	降雪量「少ない」階級の出現する確率	%

\*1 日照率40%以上の日数。日照率は、1日の日照時間を可照時間（日の出から日の入りまでの時間）で割った値。

\*2 日降水量1mm以上の日数。

\*3 日降水量10mm以上の日数。

\*4 この要素の、要素番号と予測式、予測値のカラムは全て-9999となっている。

表3 予測式の種別

番号	意味
1	日別予測式
2	期間平均予測式
3	確率ガイダンス

## 1か月予報メンバー別全球格子点値ファイルの解読（デコード）処理について

1. 解読サンプルプログラムのソースコード  
別紙参照。
2. 利用方法  
以下、解読サンプルプログラムの  
ソースコードのファイル名：dcd\_1megrib2\_sample.c  
実行ファイル名                ：dcd\_1merib2  
です。
  - (1) ccコマンドによりコンパイルしてください。その際標準算術関数を利用可能なようにライブラリをリンクしてください。  
実行例) \$ cc dcd\_1megrib2\_sample.c -lm -o dcd\_1merib2  
(dcd\_1megrib2 という実行ファイルが生成される)
    - ・ ANSI 準拠の c コンパイラでコンパイルできます。UNIX (HP-UX) 及び Linux (RedHat) での動作を確認しています。
    - ・ リトルエンディアンマシンにも対応しています。
  - (2) 次のコマンドを入力することにより、1か月予報メンバー別全球格子点値、各節の内容が端末に表示されると共に、4バイト実数形式でデコードされたデータがファイルに書き出されます。  
実行例) \$ dcd\_1megrib2 { 1か月予報メンバー別全球格子点値ファイル名 }
    - ・ デコードされたデータは、予報時間ごと、メンバーごとに、サンプルプログラムで割り付けた複数のファイルに出力されます。ファイル名は162行で使用されている関数identifydata 1310～1323行で割り付けています。
    - ・ ファイルの出力を止めたい場合は、サンプルプログラム14行目の  
#define IFOUT 1  
を  
#define IFOUT 0  
など、1以外の数字に変更してください。

UNIX は、X/Open Company Limited が独占的にライセンスしている米国ならびに他の国における登録商標です。

Linux は、Linus Torvalds の米国及びその他の国における登録商標あるいは商標です。

HP-UX は、米国 Hewlett-Packard Company のオペレーティングシステムの名称です。

RedHat は、米国 Red Hat Software, Inc. の登録商標です。

**解読サンプルプログラムのソースコード**  
 ( 1 か月予報メンバー別全球格子点値ファイル用 )

別紙

```

1  /******
2  **  Sample Program to convert  1 month  ensemble forecast GPV(GRIB2) of JMA
3  **                                  2005.07   JMA/CPD
4  **
5  **          Tested on Windows (gcc ,borlandc), Linux (gcc)
6  **          and HP-UX (native c compiler)
7  *****/
8  #include <stdio.h>
9  #include <string.h>
10 #include <math.h>
11 #include <stdlib.h>
12 #include <ctype.h>
13
14 #define IFOUT      1          /*  1 :Output decoded data          */
15 #define MASK7      0x7F
16 #define MASK1      0x80
17 #define UNDEF      -19999.
18 #define SIZEOFBUF  10000
19 #define SIZEOFBITBUF 50000
20 #define MAX_BUFF   50000    /* buffer size for gpv MAX_BUFF < 4294967295  2^32 */
21 #define MAX_IJX    20000
22     static   char      fl[40];
23     FILE      *fp,*fpg;
24     char      *fname;
25     static unsigned char  buffer[SIZEOFBUF],map[SIZEOFBITBUF],buffer2[MAX_BUFF];
26     int       ii1,ii2,jj1,jj2;
27
28     unsigned long      len,mtn;
29     unsigned long      len1,iyyyy,imm,iday,ltvn;
30     int                recno;
31     unsigned long      len2,ns2;
32     unsigned long      len3,ns3,ijmx,Ngdt,Ni,Nj,Di,Dj,mxnp,np[1000];
33     long                La1,La2,Lo1,Lo2;
34     unsigned long      len4,ns4,pc,pn,vl1,dfn,kt,lt,tys,Npdt;
35     long                nmem,tyens;
36     unsigned long      len5,ns5,ijdim,nbit,ntemplate;
37     long                E,D;
38     float               R;
39     unsigned long      len6,ns6,ifbitmap;
40     unsigned long      len7,ns7,nb,mb;
41     static float       data[MAX_IJX],wdata[MAX_IJX];
42     unsigned long      tcount=0,tcounta=0;
43     char                clv[5];
44 /*=====
45 */
46 /* TYPE DEF                                     */
47 /*=====
48 */

```

```

49  /*-----*/
50  /* TOOLS */
51  /*-----*/
52  void usage(void);
53  long longbybit(unsigned char *buf,unsigned long *pos,unsigned long nbit);
54  long convlong( unsigned char *string, int nbyte );
55  float convfloat( unsigned char *string, int nbyte );
56  long convlatlon(unsigned char *string );
57
58  /*-----*/
59  /*      used in identifydata */
60  /*-----*/
61  void fproduct(void);
62  void flevel(void);
63  void fderived(void);
64  void fhemispher(void);
65  void fperiod(void);
66
67  /*-----*/
68  /*      Identify Data (FILE NAME) */
69  /*-----*/
70  void identifydata(void);
71
72  /*-----*/
73  /*      DECODE SECTION */
74  /*-----*/
75  void dcd_sect0(void);
76  void dcd_sect1(void);
77  void dcd_sect2(void);
78  void dcd_sect3(void);
79  void dcd_sect4(void);
80  void dcd_sect5(void);
81  void dcd_sect6(void);
82  void dcd_sect7(void);
83
84  void ini2date(char mmyyyy[]);
85  void str_elem(char celem[]);
86  void out_data(void);
87
88  /*=====
89  */
90  /* MAIN PROGRAM */
91  /*=====
92  */
93  main(argc,argv)
94  int    argc;
95  char  **argv;
96  {
97      unsigned long    ns;
98      unsigned long    wkl;
99      char              wk[105],cns[5];
100      /*      Input File */
101      if(2 != argc)

```

```

102         {
103         usage();
104         exit(0);
105         }
106     if(NULL == (fp = fopen(argv[1],"rb")))
107     {
108         printf("\nCannot open FILE : %s\n",argv[1]);
109         usage();
110         exit(1);
111     }
112
113     fname=argv[1];
114     recno=0;
115     /*-----*/
116     /*  DECODE SECT 0  : Indicator Section */
117     /*-----*/
118     SECT0:
119         dcd_sect0();
120     /*-----*/
121     /*  DECODE SECT 1  : Identification Section */
122     /*-----*/
123     printf("\n<<SECTION 1>> : Identification Section \n");
124     dcd_sect1();
125     /*-----*/
126     /*  DECODE SECT 2   : (Local Use Section) */
127     /*-----*/
128     if( fread(buffer,sizeof(char), 5,fp) != 5){
129         printf("Read ERR SECT 2 !! File(%s)\n",fname);
130         exit(72);
131     }
132     len2=convlng(&buffer[0], 4);
133     ns2=convlng(&buffer[4], 1);
134     fseek(fp,-5,1);          /* Back Space 5 bytes */
135     switch(ns2){
136     case 2:
137         printf("<<SECTION 2>> : (Local Use Section)\n");
138         goto SECT2;
139     case 3:
140         printf("\n<<SECTION 3>> : Grid Definition Section\n");
141         goto SECT3;
142     default:
143         printf("ERROR in Section 8 !!");
144         printf(" len ns = %d %d\n",wkl,buffer[0]);
145         exit(99);
146     }
147     SECT2:
148         dcd_sect2();
149     /*-----*/
150     /*  DECODE SECT 3   : Grid Definition Section */
151     /*-----*/
152     printf("\n<<SECTION 3>> : Grid Definition Section\n");
153     SECT3:
154         dcd_sect3();

```

```

155  /*-----*/
156  /*  DECODE SECT 4      : Product Definition Section          */
157  /*-----*/
158  printf("¥n<<SECTION 4>> : Product Definition Section¥n");
159  SECT4:
160  dcd_sect4();
161  /* Identify Data(elm lvl date period */
162  identifydata();
163  /*-----*/
164  /*  DECODE SECT 5      : Data Representation Section          */
165  /*-----*/
166  printf("¥n<<SECTION 5>> : Data Representation Section¥n");
167  dcd_sect5();
168  /*-----*/
169  /*  DECODE SECT 6      : Bit-map Section                      */
170  /*-----*/
171  printf("¥n<<SECTION 6>> : Bit-map Section¥n");
172  dcd_sect6();
173  /*-----*/
174  /*  DECODE SECT 7      : Data Section                          */
175  /*-----*/
176  printf("¥n<<SECTION 7>> : Data Section¥n");
177  dcd_sect7();
178  if( IFOUT == 1 ) out_data();          /*  OutPut Data  */
179  /*-----*/
180  /*  DECODE SECT 8      : END or LOOP ?                          */
181  /*-----*/
182  if( fread(buffer,sizeof(char), 4,fp) != 4 ){
183  printf("Read ERR !! File(%s)¥n",argv[1]);
184  exit(70);
185  }
186  if(buffer[0] == '7' && buffer[1] == '7' && buffer[2] == '7' && buffer[3] == '7'){
187  printf("GRIB END !!");
188  goto SECT0;
189  /*exit(0);*/
190  }
191  else{
192  wkl=convlng(&buffer[0], 4);
193  if( fread(buffer,sizeof(char), 1,fp) != 1 ){
194  printf("Read ERR !! File(%s)¥n",argv[1]);
195  exit(71);
196  }
197  fseek(fp,-5,1);          /*  Back Space 5 bytes  */
198  ns=convlng(&buffer[0], 1);
199  switch(ns){
200  case 2:
201  printf("¥n-----¥n");
202  printf("Record No=%d ¥n",++recno);
203  printf("-----¥n");
204  printf("<<SECTION 2>> : (Local Use Section)¥n");
205  goto SECT2;
206  case 3:
207  printf("¥n-----¥n");

```

```

208         printf("Record No=%d %n",++recno);
209         printf("-----%n");
210         printf("%n<<SECTION 3>> : Grid Definition Section%n");
211         goto SECT3;
212     case 4:
213         printf("%n-----%n");
214         printf("Record No=%d %n",++recno);
215         printf("-----%n");
216         printf("%n<<SECTION 4>> : Product Definition Section%n");
217         goto SECT4;
218     default:
219         printf("ERROR in Section 8 !!");
220         printf(" len ns = %d %d%n",wkl,buffer[0]);
221         exit(99);
222     }
223 }
224 }
225 /*=====*/
226 /*  END MAIN()  */
227 /*=====*/
228 /*  DEFINE FUNCTIONS  */
229 /*=====*/
230 /*-----*/
231 /*  DECODE SECTION 0  */
232 /*-----*/
233 void dcd_sect0(void)
234 {
235     /*  DECODE SECT 0  */
236     /*  Check Header  */
237     if( fread(buffer,sizeof(char), 4, fp) == 0 ) exit(2);
238     while( buffer[0] != 'G' || buffer[1] != 'R' ||
239           buffer[2] != 'I' || buffer[3] != 'B'){
240         printf("Cannot Find GRIB header !!%n");
241
242         fclose(fp);exit(99);
243     }
244     printf("-----%n");
245     printf("---- GRIB FILE !! ----%n");
246     printf("-----%n");
247     fread(buffer,sizeof(char), 12, fp);
248     mtn=convlong(&buffer[2], 1);
249     printf("%n<<SECTION 0>> : Indicator Section%n");
250     printf("GRIB Master Table Number : %d%n",mtn);
251     printf("GRIB Edition Number      : %d%n",buffer[3]);
252     printf("buffer[4]= %d%n",convlong(&buffer[4],4));
253     if( convlong(&buffer[4],4) == 0){
254         len = convlong(&buffer[8], 4);
255         printf("Total length of GRIB      : %d%n",len);
256     }
257     else
258     {
259         printf("Larg Record !! Not Supported !!%n%n");
260         exit(3);

```

```

261     }
262 }
263
264 /*-----*/
265 /*  DECODE SECTION 1                                     */
266 /*-----*/
267 void dcd_sect1(void)
268 {
269     unsigned long    ns1,idcenter,icsubc;
270
271     fread(buffer,sizeof(char), 5,fp);
272
273     len1=convlong(&buffer[0], 4);
274     ns1=convlong(&buffer[4], 1);
275     fread(buffer,sizeof(char),len1-5,fp);
276     idcenter=convlong(&buffer[0], 2);
277     icsubc=convlong(&buffer[2],2);
278     ltvn=convlong(&buffer[5],1);
279     iyyyy=convlong(&buffer[7], 2);
280     imm=convlong(&buffer[9], 1);
281     iday=convlong(&buffer[10], 1);
282     printf("Length of Section 1   : %d\n",len1);
283     printf("Number of Section     : %d\n",ns1);
284     printf("Identification of centre : %d\n",idcenter);
285     printf("Identification of sub-centre : %d\n",icsubc);
286     printf("GRIB Master Tables Version Number : %d\n",buffer[4]);
287     printf("GRIB Local Tables Version Number : %d\n",buffer[5]);
288     printf("Significance of Reference Time   : %d\n",buffer[6]);
289     printf("Year                          : %d\n",iyyyy);
290     printf("Month                          : %d\n",imm);
291     printf("Day                            : %d\n",iday);
292     printf("Hour                            : %d\n",buffer[11]);
293     printf("Minute                           : %d\n",buffer[12]);
294     printf("Second                           : %d\n",buffer[13]);
295     printf("Production status of processed data : %d\n",buffer[14]);
296     printf("Type of processed data           : %d\n",buffer[15]);
297
298     printf("-----\n");
299     printf("Record No=%d \n",++recno);
300     printf("-----\n");
301 }
302
303 /*-----*/
304 /*  DECODE SECTION 2                                     */
305 /*-----*/
306 void dcd_sect2(void)
307 {
308     if( fread(buffer,sizeof(char), 5,fp) != 5 ){
309         printf("Read ERR SECT 2 !! File(%s)\n",fname);
310         exit(72);
311     }
312     len2=convlong(&buffer[0], 4);
313     ns2=convlong(&buffer[4], 1);

```

```

314     printf("Length of Section 2   : %d\n",len2);
315     printf("Number of Section    : %d\n",ns2);
316     if( len2-4 != 0 ){
317         fread(buffer,sizeof(char),len2-5,fp);
318         printf(" Local use .... Not Supported!!!");
319         exit(2);
320     }
321 }
322
323 /*-----*/
324 /*  DECODE SECTION 3                               */
325 /*-----*/
326 void dcd_sect3(void)
327 {
328     unsigned long wkl,i,ol,Ntoe;
329
330     if( fread(buffer,sizeof(char), 5,fp) != 5 ){
331         printf("Read ERR SECT 3 !! File(%s)\n",fname);
332         exit(73);
333     }
334     len3=convlng(&buffer[0], 4);
335     ns3=convlng(&buffer[4], 1);
336
337     fread(buffer,sizeof(char),len3-5,fp);
338     ijmx = convlong(&buffer[1], 4);
339     Ngdt = convlong(&buffer[7], 2);
340     ol   = convlong(&buffer[5], 1);
341     printf("Length of section           : %d\n",len3);
342     printf("Number of section            : %d\n",ns3);
343     printf("Source of grid definition      : %d\n",buffer[0]);
344     printf("Number of data points           : %d\n",ijmx);
345     printf("optional list                     : %d\n",ol);
346     printf("Interpretation of list            : %d\n",buffer[6]);
347     printf("Grid Definition Template Number : %d\n",Ngdt);
348     switch(Ngdt){
349     case 0:
350         printf("((Grid Definition Template 3.0))\n");
351         printf("Shape of the earth                   : %d\n",buffer[9]);
352         printf("Scale factor of radius of spherical earth : %d\n",buffer[10]);
353         wkl = convlong(&buffer[11], 4);
354         printf("Scaled value of radius of spherical earth : %u\n",wkl);
355         printf("Scale factor of major axis of oblate spheroid earth : %d\n",buffer[15]);
356         wkl = convlong(&buffer[16], 4);
357         printf("Scaled value of major axis of oblate spheroid earth : %u\n",wkl);
358         printf("Scale factor of minor axis of oblate spheroid earth : %d\n",buffer[20]);
359         wkl = convlong(&buffer[21], 4);
360         printf("Scaled value of minor axis of oblate spheroid earth : %u\n",wkl);
361         Ni = convlong(&buffer[25], 4);
362         Nj = convlong(&buffer[29], 4);
363         printf("number of points along a parallel         : %d\n",Ni);
364         printf("number of points along a meridian        : %d\n",Nj);
365         wkl = convlong(&buffer[33], 4);
366         printf("Basic angle of the initial production domain : %d\n",wkl);

```

```

367     wkl = convlong(&buffer[37], 4);
368     printf("Subdivisions of basic angle                : %u¥n", wkl);
369     La1 = convlong(&buffer[41], 4);
370     Lo1 = convlong(&buffer[45], 4);
371     printf("latitude of first grid point(La1)         : %d¥n", La1);
372     printf("longitude of first grid point(Lo1)        : %d¥n", Lo1);
373     printf("Resolution and component flags           : %d¥n", buffer[49]);
374     La2 = convlatlon(&buffer[50]);
375     Lo2 = convlatlon(&buffer[54]);
376     printf("latitude of last grid point(La2)         : %d¥n", La2);
377     printf("longitude of last grid point(Lo2)        : %d¥n", Lo2);
378     Di = convlong(&buffer[58], 4);
379     Dj = convlong(&buffer[62], 4);
380     printf("i direction increment(Di)                : %d¥n", Di);
381     printf("j direction increment(Dj)                : %d¥n", Dj);
382     printf("Scanning mode                            : %d¥n", buffer[66]);
383     break;
384 case 40: /* Not Checked TEST TEST TEST */
385     printf("((Grid Definition Template 3.40))¥n");
386     printf("Shape of the earth                          : %d¥n", buffer[9]);
387     printf("Scale factor of radius of spherical earth         : %d¥n", buffer[10]);
388     wkl = convlong(&buffer[11], 4);
389     printf("Scaled value of radius of spherical earth         : %d¥n", wkl);
390     printf("Scale factor of major axis of oblate spheroid earth : %d¥n", buffer[15]);
391     wkl = convlong(&buffer[16], 4);
392     printf("Scaled value of major axis of oblate spheroid earth : %d¥n", wkl);
393     printf("Scale factor of minor axis of oblate spheroid earth : %d¥n", buffer[20]);
394     wkl = convlong(&buffer[21], 4);
395     printf("Scaled value of minor axis of oblate spheroid earth : %d¥n", wkl);
396     Ni = convlong(&buffer[25], 4);
397     Nj = convlong(&buffer[29], 4);
398     printf("number of points along a parallel                 : %d¥n", Ni);
399     printf("number of points along a meridian                 : %d¥n", Nj);
400     wkl = convlong(&buffer[33], 4);
401     printf("Basic angle of the initial production domain      : %d¥n", wkl);
402     wkl = convlong(&buffer[37], 4);
403     printf("Subdivisions of basic angle                        : %d¥n", wkl);
404     La1 = convlong(&buffer[41], 4);
405     Lo1 = convlong(&buffer[45], 4);
406     printf("latitude of first grid point(La1)                 : %d¥n", La1);
407     printf("longitude of first grid point(Lo1)                : %d¥n", Lo1);
408     printf("Resolution and component flags                     : %d¥n", buffer[49]);
409     La2 = convlatlon(&buffer[50]);
410     Lo2 = convlatlon(&buffer[54]);
411     printf("latitude of last grid point(La2)                 : %d¥n", La2);
412     printf("longitude of last grid point(Lo2)                : %d¥n", Lo2);
413     Di = convlong(&buffer[58], 4);
414     Ntoe = convlong(&buffer[62], 4);
415     printf("i direction increment(Di)                          : %d¥n", Di);
416     printf("Number of parallels from pole to equator(N)        : %d¥n", Ntoe);
417     printf("Scanning mode                                      : %d¥n", buffer[66]);
418     printf("Number of points along each grid line ¥n");
419     printf("    Line    Points¥n");

```

```

420         mxnp=0;
421         for(i=0 ;i < Nj ;++i)
422             {
423                 np[i]=convlong(&buffer[67+i*2], ol );
424                 if(np[i] > mxnp) mxnp=np[i];
425             }
426         break;
427     default      :
428         printf("This Grid is Not Supported !!");
429         exit(10);
430     }
431 }
432
433 /*-----*/
434 /*  DECODE SECTION 4                                     */
435 /*-----*/
436 void dcd_sect4(void)
437 {
438     unsigned long    noc,tcut,svl1,svl2;
439     unsigned long    iyyyye,imme,idaye,Nt,nmiss,tinc;
440     unsigned long    i,latn,lats,lone,lonw,Nc,vstd,vdist;
441     char             sfl1;
442
443     if( fread(buffer,sizeof(char), 5,fp) != 5 ){
444         printf("Read ERR SECT 4 !! File(%s)¥n",fname);
445         exit(74);
446     }
447     len4=convlong(&buffer[0], 4);
448     ns4=convlong(&buffer[4], 1);
449
450     fread(buffer,sizeof(char),len4-5,fp);
451     noc = convlong(&buffer[0], 2);
452     Npdt= convlong(&buffer[2], 2);
453     printf("Length of section 4                : %d¥n",len4);
454     printf("Number of section                    : %d¥n",ns4);
455     printf("Number of coordinates values after Template : %d¥n",noc);
456     printf("Product Definition Template Number          : %d¥n",Npdt);
457     printf("(( Product Definition Template 4.%d ))¥n",Npdt);
458     switch(Npdt){
459
460     case    1:                /*  PDT4. 1  */
461         pc    = convlong(&buffer[4], 1);
462         pn    = convlong(&buffer[5], 1);
463         tcut  = convlong(&buffer[9], 2);
464         kt    = convlong(&buffer[13], 4);
465         tys   = convlong(&buffer[17], 1);
466         sfl1  = buffer[18];
467         svl1  = convlong(&buffer[19], 4);
468         vl1   = svl1*pow(10.0,(double)sfl1);
469         svl2  = convlong(&buffer[25], 4);
470         tyens = convlong(&buffer[29], 1);
471         nmem  = convlong(&buffer[30], 1);
472         /*iyyyye = convlong(&buffer[32], 2);

```

```

473     imme   = convlong(&buffer[34], 1);
474     idaye  = convlong(&buffer[35], 1);
475     Nt     = convlong(&buffer[39], 1);
476     nmiss  = convlong(&buffer[40], 4);*/
477     printf("Parameter category                : %d¥n",pc);
478     printf("Parameter number                 : %d¥n",pn);
479     printf("Type of generating process       : %d¥n",buffer[6]);
480     printf("Background generating process identifier : %d¥n",buffer[7]);
481     printf("Forecast generating process identifier : %d¥n",buffer[8]);
482     printf("Hours after reference time of data cut-off : %d¥n",tcut);
483     printf("Minutes after reference time of data cut-off : %d¥n",buffer[11]);
484     printf("Indicator of unit of time range       : %d¥n",buffer[12]);
485     printf("Forecast time in units defined by octet 18 : %d¥n",kt);
486     printf("Type of first fixed surface           : %d¥n",tys);
487     printf("Scale factor of first fixed surface       : %u¥n",sfl1);
488     printf("Scaled value of first fixed surface      : %u¥n",svl1);
489     printf("Type of second fixed surface             : %d¥n",buffer[23]);
490     printf("Scale factor of second fixed surface     : %d¥n",buffer[24]);
491     printf("Scaled value of second fixed surface     : %u¥n",svl2);
492     printf("Type of ensemble forecast               : %d¥n",buffer[29]);
493     printf("Perturbation number                     : %d¥n",buffer[30]);
494     printf("Number of forecasts in ensemble         : %d¥n",buffer[31]);
495     break;
496     case    2:                               /* PDT4.2 */
497         pc   = convlong(&buffer[4], 1);
498         pn   = convlong(&buffer[5], 1);
499         tcut = convlong(&buffer[9], 2);
500         kt   = convlong(&buffer[13], 4);
501         tys  = convlong(&buffer[17], 1);
502         sfl1 = buffer[18];
503         svl1 = convlong(&buffer[19], 4);
504         vl1  = svl1*pow(10.0,(double)sfl1);
505         svl2 = convlong(&buffer[25], 4);
506         dfn  = convlong(&buffer[29], 1);
507         nmem = -1;
508         printf("Parameter category                : %d¥n",pc);
509         printf("Parameter number                 : %d¥n",pn);
510         printf("Type of generating process       : %d¥n",buffer[6]);
511         printf("Background generating process identifier : %d¥n",buffer[7]);
512         printf("Forecast generating process identifier : %d¥n",buffer[8]);
513         printf("Hours after reference time of data cut-off : %d¥n",tcut);
514         printf("Minutes after reference time of data cut-off : %d¥n",buffer[11]);
515         printf("Indicator of unit of time range       : %d¥n",buffer[12]);
516         printf("Forecast time in units defined by octet 18 : %d¥n",kt);
517         printf("Type of first fixed surface           : %u¥n",tys);
518         printf("Scale factor of first fixed surface       : %u¥n",sfl1);
519         printf("Scaled value of first fixed surface      : %u¥n",svl1);
520         printf("Type of second fixed surface             : %d¥n",buffer[23]);
521         printf("Scale factor of second fixed surface     : %d¥n",buffer[24]);
522         printf("Scaled value of second fixed surface     : %u¥n",svl2);
523         printf(" Derived forecast (see Code Table 4.7) : %u¥n",dfn);
524         printf("Number of forecasts in ensemble         : %d¥n",buffer[30]);
525         break;

```

```

526     case    11:                /* PDT4.11 */
527         pc    = convlong(&buffer[4], 1);
528         pn    = convlong(&buffer[5], 1);
529         tcut  = convlong(&buffer[9], 2);
530         kt    = convlong(&buffer[13], 4);
531         tys   = convlong(&buffer[17], 1);
532         sfl1  = buffer[18];
533         svl1  = convlong(&buffer[19], 4);
534         vl1   = svl1*pow(10.0,(double)sfl1);
535         svl2  = convlong(&buffer[25], 4);
536         tyens = convlong(&buffer[29], 1);
537         nmem  = convlong(&buffer[30], 1);
538         iyyyye = convlong(&buffer[32], 2);
539         imme  = convlong(&buffer[34], 1);
540         idaye = convlong(&buffer[35], 1);
541         Nt    = convlong(&buffer[39], 1);
542         nmiss = convlong(&buffer[40], 4);
543         printf("Parameter category           : %d\n",pc);
544         printf("Parameter number            : %d\n",pn);
545         printf("Type of generating process       : %d\n",buffer[6]);
546         printf("Background generating process identifier : %d\n",buffer[7]);
547         printf("Forecast generating process identifier  : %d\n",buffer[8]);
548         printf("Hours after reference time of data cut-off : %d\n",tcut);
549         printf("Minutes after reference time of data cut-off : %d\n",buffer[11]);
550         printf("Indicator of unit of time range          : %d\n",buffer[12]);
551         printf("Forecast time in units defined by octet 18 : %d\n",kt);
552         printf("Type of first fixed surface              : %u\n",tys);
553         printf("Scale factor of first fixed surface      : %u\n",sfl1);
554         printf("Scaled value of first fixed surface      : %u\n",svl1);
555         printf("Type of second fixed surface             : %d\n",buffer[23]);
556         printf("Scale factor of second fixed surface     : %d\n",buffer[24]);
557         printf("Scaled value of second fixed surface     : %u\n",svl2);
558         printf("Type of ensemble forecast               : %d\n",buffer[29]);
559         printf("Perturbation number                     : %d\n",buffer[30]);
560         printf("Number of forecasts in ensemble         : %d\n",buffer[31]);
561         printf("Year                                     : %d\n",iyyyye);
562         printf("Month                                    : %d\n",imme);
563         printf("Day                                       : %d\n",idaye);
564         printf("Hour                                     : %d\n",buffer[36]);
565         printf("Minut                                    : %d\n",buffer[37]);
566         printf("Second                                   : %d\n",buffer[38]);
567         printf("n - Number of time range                 : %d\n",Nt);
568         printf("Total number of data values missing      : %d\n",nmiss);
569         for(i=0 ; i < Nt ; ++i){
570             lt    = convlong(&buffer[47+i*12], 4);
571             tinc  = convlong(&buffer[52+i*12], 4);
572             printf("Statistical process used to calculate .... : %d\n",buffer[44+i*12]);
573             printf("Type of time increment                    : %d\n",buffer[45+i*12]);
574             printf("Indicator of unit of time for time range   : %d\n",buffer[46+i*12]);
575             printf("Length of the time range                  : %d\n",lt);
576             printf("Indicator of unit of time for the increment : %d\n",buffer[51+i*12]);
577             printf("Time increment between successive fields   : %d\n",tinc);
578         }

```

```

579         break;
580     case 12: /* PDT4.12 */
581         pc = convlong(&buffer[4], 1);
582         pn = convlong(&buffer[5], 1);
583         tcut = convlong(&buffer[9], 2);
584         kt = convlong(&buffer[13], 4);
585         tys = convlong(&buffer[17], 1);
586         sfl1 = buffer[18];
587         svl1 = convlong(&buffer[19], 4);
588         vl1 = svl1*pow(10.0,(double)sfl1);
589         svl2 = convlong(&buffer[25], 4);
590         dfn = convlong(&buffer[29], 1);
591         nmem = -1;
592         iyyyye = convlong(&buffer[31], 2);
593         imme = convlong(&buffer[33], 1);
594         idaye = convlong(&buffer[34], 1);
595         Nt = convlong(&buffer[38], 1);
596         nmiss = convlong(&buffer[39], 4);
597         printf("Parameter category : %d\n",pc);
598         printf("Parameter number : %d\n",pn);
599         printf("Type of generating process : %d\n",buffer[6]);
600         printf("Background generating process identifier : %d\n",buffer[7]);
601         printf("Forecast generating process identifier : %d\n",buffer[8]);
602         printf("Hours after reference time of data cut-off : %d\n",tcut);
603         printf("Minutes after reference time of data cut-off : %d\n",buffer[11]);
604         printf("Indicator of unit of time range : %d\n",buffer[12]);
605         printf("Forecast time in units defined by octet 18 : %d\n",kt);
606         printf("Type of first fixed surface : %u\n",tys);
607         printf("Scale factor of first fixed surface : %u\n",sfl1);
608         printf("Scaled value of first fixed surface : %u\n",svl1);
609         printf("Type of second fixed surface : %d\n",buffer[23]);
610         printf("Scale factor of second fixed surface : %d\n",buffer[24]);
611         printf("Scaled value of second fixed surface : %u\n",svl2);
612         printf(" Derived forecast (see Code Table 4.7) : %u\n",dfn);
613         printf("Number of forecasts in ensemble : %d\n",buffer[30]);
614         printf("Year : %d\n",iyyyye);
615         printf("Month : %d\n",imme);
616         printf("Day : %d\n",idaye);
617         printf("Hour : %d\n",buffer[35]);
618         printf("Minut : %d\n",buffer[36]);
619         printf("Second : %d\n",buffer[37]);
620         printf("n - Number of time range : %d\n",Nt);
621         printf("Total number of data values missing : %d\n",nmiss);
622         for(i=0 ; i < Nt ; ++i){
623             lt = convlong(&buffer[46+i*12], 4);
624             tinc = convlong(&buffer[51+i*12], 4);
625             printf("Statistical process used to calculate .... : %d\n",buffer[43+i*12]);
626             printf("Type of time increment : %d\n",buffer[44+i*12]);
627             printf("Indicator of unit of time for time range : %d\n",buffer[45+i*12]);
628             printf("Length of the time range : %d\n",lt);
629             printf("Indicator of unit of time for the increment : %d\n",buffer[50+i*12]);
630             printf("Time increment between successive fields : %d\n",tinc);
631         }

```

```

632         break;
633     case 13:             /* PDT4.13 */
634         pc = convlong(&buffer[4], 1);
635         pn = convlong(&buffer[5], 1);
636         tcut = convlong(&buffer[9], 2);
637         kt = convlong(&buffer[13], 4);
638         tys = convlong(&buffer[17], 1);
639         sfl1 = buffer[18];
640         svl1 = convlong(&buffer[19], 4);
641         vl1 = svl1*pow(10.0,(double)sfl1);
642         svl2 = convlong(&buffer[25], 4);
643         dfn = convlong(&buffer[29], 1);
644         nmem = -1;
645         latn = convlong(&buffer[36], 4);
646         lats = convlong(&buffer[40], 4);
647         lone = convlong(&buffer[44], 4);
648         lonw = convlong(&buffer[48], 4);
649         Nc = convlong(&buffer[52], 1);
650         vstd = convlong(&buffer[54], 4);
651         vdist = convlong(&buffer[59], 4);
652         iyyyye = convlong(&buffer[63], 2);
653         imme = convlong(&buffer[65], 1);
654         idaye = convlong(&buffer[66], 1);
655         Nt = convlong(&buffer[70], 1);
656         nmiss = convlong(&buffer[71], 4);
657         printf("Parameter category                : %d\n",pc);
658         printf("Parameter number                  : %d\n",pn);
659         printf("Type of generating process                : %d\n",buffer[6]);
660         printf("Background generating process identifier    : %d\n",buffer[7]);
661         printf("Forecast generating process identifier      : %d\n",buffer[8]);
662         printf("Hours after reference time of data cut-off  : %d\n",tcut);
663         printf("Minutes after reference time of data cut-off : %d\n",buffer[11]);
664         printf("Indicator of unit of time range             : %d\n",buffer[12]);
665         printf("Forecast time in units defined by octet 18   : %d\n",kt);
666         printf("Type of first fixed surface                  : %u\n",tys);
667         printf("Scale factor of first fixed surface          : %u\n",sfl1);
668         printf("Scaled value of first fixed surface          : %u\n",svl1);
669         printf("Type of second fixed surface                 : %d\n",buffer[23]);
670         printf("Scale factor of second fixed surface         : %d\n",buffer[24]);
671         printf("Scaled value of second fixed surface         : %u\n",svl2);
672         printf("Derived forecast (see Code Table 4.7)       : %u\n",dfn);
673         printf("Number of forecasts in the ensemble (N)     : %d\n",buffer[30]);
674         printf("Cluster identifier                           : %d\n",buffer[31]);
675         printf("Number of cluster to which the high resolution control
676 belongs : %d\n",buffer[32]);
677         printf("Number of cluster to which the low resolution control
678 belongs : %d\n",buffer[33]);
679         printf("Total number of clusters                     : %d\n",buffer[34]);
680         printf("Clustering method (see Code Table 4.8)      : %d\n",buffer[35]);
681         printf("Northern latitude of cluster domain         : %d\n",latn);
682         printf("Southern latitude of cluster domain         : %d\n",lats);
683         printf("Eastern longitude of cluster domain         : %d\n",lone);
684         printf("Western longitude of cluster domain         : %d\n",lonw);

```

```

685     printf("Number of forecasts in the cluster           : %d\n",Nc);
686     printf("Scale factor of standard deviation in the cluster           : %d\n",buffer[53]);
687     printf("Scaled value of standard deviation in the cluster           : %u\n",vstd);
688     printf("Scale factor of distance of the cluster from ensemble mean : %d\n",buffer[58]);
689     printf("Scaled value of distance of the cluster from ensemble mean : %u\n",vdist);
690     printf("Year                : %d\n",iyyyye);
691     printf("Month               : %d\n",imme);
692     printf("Day                 : %d\n",idaye);
693     printf("Hour                : %d\n",buffer[67]);
694     printf("Minut               : %d\n",buffer[68]);
695     printf("Second              : %d\n",buffer[69]);
696     printf("n - Number of time range      : %d\n",Nt);
697     printf("Total number of data values missing           : %d\n",nmiss);
698     for(i=0 ; i < Nt ; ++i){
699         lt    = convlong(&buffer[78+i*12], 4);
700         tinc  = convlong(&buffer[83+i*12], 4);
701         printf("Statistical process used to calculate .... : %d\n",buffer[75+i*12]);
702         printf("Type of time increment                   : %d\n",buffer[76+i*12]);
703         printf("Indicator of unit of time for time range : %d\n",buffer[77+i*12]);
704         printf("Length of the time range                 : %d\n",lt);
705         printf("Indicator of unit of time for the increment : %d\n",buffer[82+i*12]);
706         printf("Time increment between successive fields : %d\n",tinc);
707     }
708     printf("List of Nc ensemble forecast numbers :");
709     for(i=0 ; i < Nc ; ++i){
710         printf(" %2d",buffer[74+Nt*12+i+1]);
711     }
712     printf("\n");
713     break;
714 default:
715     printf("This Product Definition Template is Not Supported !!");
716     exit(11);
717 }
718 }
719 /*-----*/
720 /*  DECODE SECTION 5                                     */
721 /*-----*/
722 void dcd_sect5(void)
723 {
724     fread(buffer,sizeof(char), 5,fp);
725     len5=convlong(&buffer[0], 4);
726     ns5=convlong(&buffer[4], 1);
727     fread(buffer,sizeof(char),len5-5,fp);
728     ijdin = convlong(&buffer[0], 4);
729     ntemplate = convlong(&buffer[4], 2);
730
731     printf("Length of section in octets           : %d\n",len5);
732     printf("Number of section                   : %d\n",ns5);
733     printf("Number of data points               : %d\n",ijdin);
734     printf("Data Representation Template Number : %d\n",ntemplate);
735     switch(ntemplate){
736     case 0:
737         R = convfloat(&buffer[6], 4);

```

```

738     E = convlong(&buffer[10], 2);
739         if(E > 32768) E = (E - 32768)*(-1);
740     D = convlong(&buffer[12], 2);
741         if(D > 32768) D = (D - 32768)*(-1);
742     nbit = convlong(&buffer[14], 1);
743     printf("Reference value (R)                : %f¥n",R);
744     printf("Binary scale factor (E)            : %d¥n",E);
745     printf("Decimal scale factor (D)          : %d¥n",D);
746     printf("Number of bits .....           : %d¥n",nbit);
747     printf("Type of original field values    : %d¥n",buffer[15]);
748     break;
749     default:
750         printf("This Data Representation Template is Not Supported!!");
751         exit(12);
752     }
753 }
754
755 /*-----*/
756 /*  DECODE SECTION 6                                     */
757 /*-----*/
758 void dcd_sect6(void)
759 {
760     int    i;
761
762     fread(buffer,sizeof(char), 5,fp);
763     len6=convlong(&buffer[0], 4);
764     ns6=convlong(&buffer[4], 1);
765     fread(buffer,sizeof(char), 1,fp);
766     ifbitmap = convlong(&buffer[0], 1);
767     printf("Length of section in octets        : %d¥n",len6);
768     printf("Number of section                    : %d¥n",ns6);
769     printf("Bit-map indicator                      : %d¥n",ifbitmap);
770     for( i = 0 ; i < SIZEOFBITBUF ;++i){
771         map[i]=0xFF;
772     }
773     switch(ifbitmap){
774     case    0:
775         if ( len6-6 != fread(map,sizeof(char),len6-6,fp)){
776             printf("Read ERR SECT 6-1 !! File(%s)¥n",fname);
777             exit(100);
778         }
779         break;
780     case    255:
781         printf("No BITMAP !!¥n");
782         break;
783     default:
784         printf("This Bit-map indicator is Not Supportedt!!");
785         exit(13);
786     }
787 }
788
789 /*-----*/
790 /*  DECODE SECTION 7                                     */

```

```

791  /*-----*/
792  void dcd_sect7(void)
793  {
794      unsigned char    msk_bit=MASK1;
795      unsigned long    posi,lgpv,i;
796
797
798      fread(buffer,sizeof(char), 5,fp);
799      len7=convlong(&buffer[0], 4);
800      ns7=convlong(&buffer[4], 1);
801      printf("Length of section in octets           : %d\n",len7);
802      printf("Number of section                   : %d\n",ns7);
803      fread(buffer2,sizeof(char),len7-5,fp);
804      printf("ntemplate=%d\n",ntemplate);
805      switch(ntemplate){
806          case 0:
807
808              posi = 0;
809              for( i = 0 ;i < ijmx ;i++ ){
810                  nb = i / 8;
811                  mb = i % 8;
812                  if( 0 == (map[nb] & ( msk_bit >> mb )))){
813                      data[i]=UNDEF;
814                  }
815                  else{
816                      lgpv=longbybit(buffer2,&posi,nbit);
817                      data[i] = (R + lgpv*pow(2.0,(double)E))/pow(10.0,(double)D);
818                  }
819              }
820
821              printf("Check DATA !!    \n");
822              printf(" No    value \n");
823              for( i = 0; i < 10 ; ++i){
824                  printf("  %5d    %f\n",i+1,data[i]);
825              }
826              printf("\n\n");
827              for( i = ijmx-10; i < ijmx ; ++i){
828                  printf("  %5d    %f\n",i+1,data[i]);
829              }
830
831              break;
832          default:
833              printf("Unsupported Template !!");
834              exit(14);
835      }
836  }
837
838  /*-----*/
839  /*  OUT_DATA                                     */
840  /*-----*/
841  void out_data(void)
842  {
843      int          itot;

```

```

844 unsigned int    i,j,ii,flen;
845 FILE           *fpc1;
846 char           flc1[30];
847 char           celem[100],mmyyyy[7];
848 float          fla1,fdi,flo1,fdj;
849
850 if(( fpg = fopen(fl,"wb")) == NULL ){
851     printf("Stop, Create File(%s)¥n",fl);
852     exit(99);
853 }
854
855 switch(Ngdt)
856 {
857     case 0:
858         if( fwrite(data,4,ijmx,fpg) == ijmx )
859             {
860                 /* Out Put GrADS Control File */
861                 strcpy(fl,flc1);
862                 flen=strlen(fl);
863                 flc1[flen-4]='.';flc1[flen-3]='c';flc1[flen-2]='t';
864                 flc1[flen-1]='l';flc1[flen]=0x00;
865                 if(( fpc1 = fopen(fl,"wt")) == NULL ){
866                     printf("Stop, Create File(%s)¥n",flc1);
867                     exit(98);
868                 }
869                 fla1=(-1)*(float)La1/1000000.0 ;fdi=(float)Di/1000000.0;
870                 flo1=(float)Lo1/1000000.0 ;fdj=(float)Dj/1000000.0;
871
872                 ini2date(mmyyyy);
873                 str_elem(celem);
874                 tcount=1;
875
876                 fprintf(fpc1,"dset ^%s¥n",fl);
877                 fprintf(fpc1,"undef -19999.¥n");
878                 fprintf(fpc1,"xdef %d linear %f %f¥n",Ni,flo1,fdi);
879                 fprintf(fpc1,"ydef %d linear %f %f¥n",Nj,fla1,fdj);
880                 fprintf(fpc1,"zdef 1 linear %s 1 ¥n",clvl);
881                 fprintf(fpc1,"tdef %d linear 01%s 1mo¥n",tcount,mmyyyy);
882                 fprintf(fpc1,"vars 1 ¥n");
883                 fprintf(fpc1,"%s¥n",celem);
884                 fprintf(fpc1,"endvars¥n");
885                 fprintf(fpc1,"options yrev¥n");
886                 printf("Close Control File= %s ¥n",flc1);
887                 if ( fclose(fpc1) != 0 ){
888                     printf("Close ERR !! File(%s)¥n",flc1);
889                     exit(96);
890                 }
891             }
892     else{
893         printf("Write ERR !! File(%s)¥n",fl);
894         exit(98);
895     }
896     break;

```

```

897     default:
898         printf(" This GDT is not supported !!  Ngdt= %d ¥n",Ngdt);
899         exit(95);
900     }
901     printf("Close GrADS File= %s ¥n",fl);
902     if ( fclose(fpg) != 0 ){
903         printf("Close ERR !! File(%s)¥n",fl);
904         exit(97);
905     }
906 }
907 /*=====*/
908 void ini2date(char mmyyyy[])
909 {
910     unsigned long yyyy1,mm1;
911     char   yyyy[5],mm[3];
912
913     if    ( imm == 12){ strcpy(mm,"Jan");yyyy1=iyyyy+1;}
914     else if( imm ==  1){ strcpy(mm,"Feb");yyyy1=iyyyy  ;}
915     else if( imm ==  2){ strcpy(mm,"Mar");yyyy1=iyyyy  ;}
916     else if( imm ==  3){ strcpy(mm,"Apr");yyyy1=iyyyy  ;}
917     else if( imm ==  4){ strcpy(mm,"May");yyyy1=iyyyy  ;}
918     else if( imm ==  5){ strcpy(mm,"Jun");yyyy1=iyyyy  ;}
919     else if( imm ==  6){ strcpy(mm,"Jul");yyyy1=iyyyy  ;}
920     else if( imm ==  7){ strcpy(mm,"Aug");yyyy1=iyyyy  ;}
921     else if( imm ==  8){ strcpy(mm,"Sep");yyyy1=iyyyy  ;}
922     else if( imm ==  9){ strcpy(mm,"Oct");yyyy1=iyyyy  ;}
923     else if( imm == 10){ strcpy(mm,"Nov");yyyy1=iyyyy  ;}
924     else if( imm == 11){ strcpy(mm,"Dec");yyyy1=iyyyy  ;}
925     else{ printf(" Initial month is strange!! imm=%d¥n",imm); exit(55);}
926
927     sprintf(mmyyyy,"%s%04d",mm,yyyy1);
928 }
929
930 void str_elem(char celem[])
931 {
932     char      elm[5],lvl[5],spr[5],comm[70],wstr[10];
933     char      sprcomm[20];
934     unsigned long vl1hpa;
935     unsigned int  i,len1,len2,len3,len4,len5,len6;
936
937     sprcomm[0]=0x00;
938     elm[0]=0x00;lvl[0]=0x00;spr[0]=0x00;
939     vl1hpa=vl1*0.01;
940
941     /*----- LEVEL -----*/
942     if    ( tys ==  1 ) {strcpy(lvl,"surf")      ;strcpy(clvl,"1000");}
943     else if( tys == 101 ){strcpy(lvl,"sea")      ;strcpy(clvl,"1000");}
944     else if( tys == 103 ){strcpy(lvl,"2m")      ;strcpy(clvl,"1000");}
945     else if( tys == 100 ){sprintf(lvl,"%01d",vl1hpa);sprintf(clvl,"%3d",vl1hpa);}
946     else  {printf(" This level is Not Supported !! tys %d¥n",tys);
947             exit(43);}
948     /*----- SPREAD ? -----*/
949     if ( dfn == 4 ){

```

```

950     strcpy(spr,"s");strcpy(sprcomm,"spread ");
951     }
952     else if( dfn == 5 ){
953         strcpy(spr,"I");strcpy(sprcomm,"larg anomaly index ");
954     }
955
956     /*----- ELEMENT -----*/
957     if( mtn == 0)
958     {
959         if ( pc == 0 && pn == 0 ){
960             if( tys != 100 ){
961                 strcpy(elm,"t") ;sprintf(comm,"Temperature %s at %s [K]",sprcomm,lv1);}
962             else{
963                 strcpy(elm,"t") ;sprintf(comm,"Temperature at %s hPa [K]",lv1);}
964             }
965         else if( pc == 0 && pn == 9 ){
966             if( tys != 100 ){
967                 strcpy(elm,"t") ;strcpy(spr,"a");
968                 sprintf(comm,"Temprature anoaly at %s hPa [K]",lv1);}
969             else{
970                 strcpy(elm,"t") ;strcpy(spr,"a");
971                 sprintf(comm,"Temprature anoaly at %s [K]",lv1);}
972             }
973         else if( pc == 1 && pn == 0 ){
974             strcpy(elm,"q") ;
975             sprintf(comm,"Specific humidity at %s hPa [kg/kg]",lv1);}
976         else if( pc == 1 && pn == 1 ){
977             strcpy(elm,"rh") ;
978             sprintf(comm,"Relative humidity %s at %s hPa [%%]",sprcomm,lv1);}
979         else if( pc == 1 && pn == 8 ){
980             strcpy(elm,"rr") ;
981             sprintf(comm,"Total precipitaion %s [mm]",sprcomm);}
982         else if( pc == 1 && pn == 210 ){
983             strcpy(elm,"rrd") ;lv1[0]=0x00;
984             sprintf(comm,"Daily mean precipitaion %s [mm]",sprcomm);}
985         else if( pc == 1 && pn == 211 ){
986             strcpy(elm,"rrd") ;strcpy(spr,"a");lv1[0]=0x00;
987             sprintf(comm,"Daily mean precipitaion anomaly [mm]");}
988         else if( pc == 1 && pn == 212 ){
989             strcpy(elm,"q") ;strcpy(spr,"a");
990             sprintf(comm,"Specific humidity anomaly at %s hPa [kg/kg]",lv1);}
991         else if( pc == 1 && pn == 213 ){
992             strcpy(elm,"rh") ;strcpy(spr,"a");
993             sprintf(comm,"Relative humidity anomaly at %s hPa [%%]",lv1);}
994         else if( pc == 2 && pn == 2 ){
995             strcpy(elm,"u") ;
996             sprintf(comm,"u wind %s at %s hPa [m/s]",sprcomm,lv1);}
997         else if( pc == 2 && pn == 3 ){
998             strcpy(elm,"v") ;
999             sprintf(comm,"v wind %s at %s hPa [m/s]",sprcomm,lv1);}
1000        else if( pc == 2 && pn == 4 ){
1001            strcpy(elm,"psi") ;
1002            sprintf(comm,"Stream function at %s hPa [m^2/s]",lv1);}

```

```

1003     else if( pc == 2 && pn == 5 ){
1004         strcpy(elm,"chi") ;
1005         sprintf(comm,"Velocity potential at %s hPa [m^2/s]",lvl);}
1006     else if( pc == 2 && pn ==210){
1007         strcpy(elm,"u") ;strcpy(spr,"a");
1008         sprintf(comm,"u wind anomaly at %s hPa [m/s]",lvl);}
1009     else if( pc == 2 && pn ==211){
1010         strcpy(elm,"v") ;strcpy(spr,"a");
1011         sprintf(comm,"v wind anomaly at %s hPa [m/s]",lvl);}
1012     else if( pc == 3 && pn == 0 ){
1013         if(tys != 100){
1014             strcpy(elm,"p") ;
1015             sprintf(comm,"Pressure %s at %s [Pa]",sprcomm,lvl);}
1016         else{
1017             strcpy(elm,"p") ;
1018             sprintf(comm,"Pressure %s at %s hPa [Pa]",sprcomm,lvl);}
1019         }
1020     else if( pc == 3 && pn == 1 ){
1021         strcpy(elm,"p");
1022         sprintf(comm,"Sea level pressure %s [Pa]",sprcomm);}
1023     else if( pc == 3 && pn == 5 ){
1024         strcpy(elm,"z") ;
1025         sprintf(comm,"Geopotential height %s at %s hPa [gpm]",sprcomm,lvl);}
1026     else if( pc == 3 && pn == 8 ){
1027         if( tys != 100 ){
1028             strcpy(elm,"p") ; strcpy(spr,"a");
1029             sprintf(comm,"Pressure anomaly at %s [Pa]",lvl);}
1030         else{
1031             strcpy(elm,"p") ; strcpy(spr,"a");
1032             sprintf(comm,"Pressure anomaly at %s hPa [Pa]",lvl);}
1033         }
1034     else if( pc == 3 && pn == 9 ){
1035         strcpy(elm,"z") ;strcpy(spr,"a");
1036         sprintf(comm,"Geopotential height anomaly at %s hPa [gpm]",lvl);}
1037     else {
1038         printf(" This element Not Supported !! mtn,pc,pn %d,%d,%d¥n",mtn,pc,pn); exit(40);}
1039     }
1040     else if(mtn == 10 )
1041     {
1042         if ( pc == 3 && pn == 0 ){
1043             strcpy(elm,"sst") ;lvl[0]=0x00;
1044             sprintf(comm,"Sea surface temperature [K]");}
1045         else if( pc == 3 && pn ==192){
1046             strcpy(elm,"sst");strcpy(spr,"a");lvl[0]=0x00;
1047             sprintf(comm,"Sea surface temperature anomaly [K]");}
1048         else {
1049             printf(" This element Not Supported !! mtn,pc,pn %d,%d,%d¥n",mtn,pc,pn); exit(41);}
1050         }
1051     else
1052     {
1053         printf(" This element Not Supported !! mtn,pc,pn %d,%d,%d¥n",mtn,pc,pn); exit(42);
1054         }
1055     /*----- length of string & copy string -----*/

```

```

1056     strcpy(wstr,"1 99 ** ");
1057
1058     len1=strlen(elm);
1059     len2=len1 + strlen(lvl);
1060     len3=len2 + strlen(spr);
1061     len4=8 ;
1062     len5=len4 + strlen(wstr);
1063     len6=len5 + strlen(comm);
1064
1065     for (i = 0 ; i < len1 ;++i)celem[i]=elm[i];
1066     for (i = len1; i < len2 ;++i)celem[i]=lvl[i-len1];
1067     for (i = len2; i < len3 ;++i)celem[i]=spr[i-len2];
1068     for (i = len3; i < len4 ;++i)celem[i]=' ';
1069     for (i = len4; i < len5 ;++i)celem[i]=wstr[i-len4];
1070     for (i = len5; i < len6 ;++i)celem[i]=comm[i-len5];
1071     celem[len6]=0x00;
1072
1073     /** printf("len1,len2,len3,len4,len5,len6=%d,%d,%d,%d,%d,%d\n",len1,len2,len3,len4,len5,len6);
1074         printf(" elm = %s\n",elm);
1075         printf(" lvl = %s\n",lvl);
1076         printf(" apr = %s\n",spr);
1077         printf(" wstr = %s\n",wstr);
1078         printf(" comm = %s\n",comm);
1079         printf(" ELEM= %s\n",celem);
1080     **/
1081 }
1082
1083
1084 /*=====*/
1085 /*  TOOLS                                     */
1086 /*=====*/
1087
1088 void usage(void)
1089 {
1090     puts("\n");
1091     puts("Usage : GRIB2 <FILE-NAME>");
1092 }
1093
1094 long longbybit(unsigned char *buf,unsigned long *pos,unsigned long nbit)
1095 {
1096     char    i,ii;
1097     long    pos8,mg8,rem8,nbytes,remnbit;
1098     long    lout=0;
1099     long    lbig;
1100     unsigned char    *buf2,*cbig;
1101     unsigned char    mask,maskremn,full=0xff;
1102
1103     lbig=1;
1104     cbig = (unsigned char *)&lbig;
1105
1106     buf2 = (unsigned char *)&lout;
1107     mg8  = *pos % 8;
1108     remnbit= nbit % 8;

```

```

1109
1110     if( (*pos + nbit) % 8 == 0 ) rem8 = 0;
1111     else                               rem8 = 8 - (( *pos + nbit ) % 8 );
1112
1113     pos8 = ( *pos + nbit + rem8 ) / 8 - 1;
1114
1115     nbytes = ( nbit ) / 8;
1116     mask   = full >>  mg8 + rem8;
1117     for( i = 0; i < nbytes ; i++){
1118         if( 1 == cbig[3] & 0x01 ) ii = 3 - i;
1119         else                               ii = i   ;
1120         buf2[ii] = ( buf[pos8 - i] >> rem8
1121                   | buf[pos8 - i - 1] << 8 - rem8 );
1122     }
1123
1124     if( 1 == cbig[3] & 0x01 ) ii = 3 - nbytes;
1125     else                               ii = nbytes ;
1126     if( remnbit <= 8 - rem8 ){
1127         maskremn = full >> 8 - remnbit;
1128         buf2[ii] = ( buf[pos8 - nbytes] >> rem8 & maskremn );
1129     }
1130     else{
1131         maskremn = full >> 8 - remnbit & full << 8 - rem8 ;
1132         buf2[ii] = ( buf[pos8 - nbytes] >> rem8
1133                   | buf[pos8 - nbytes -1] << 8 - rem8 & maskremn );
1134     }
1135     *pos = *pos + nbit;
1136     return(lout);
1137 }
1138
1139 long convlong( unsigned char *string, int nbyte )
1140 {
1141     int          i,ii;
1142     long         numb=0;
1143     long         lbig;
1144     unsigned char *chrwrk,*cbig;
1145
1146     lbig=1;
1147     cbig = (unsigned char *)&lbig;
1148     /*if( 1 == cbig[3] & 0x01 ) printf("BIG ENDIAN !!");*/
1149     chrwrk = (unsigned char *)&numb;
1150     for( i = 0; i < nbyte; i++ ) {
1151         if( 1 == cbig[3] & 0x01 ) ii = 4 - nbyte + i;
1152         else                               ii = nbyte - 1 - i;
1153
1154         chrwrk[ii] = string[i];
1155     }
1156     return( numb );
1157 }
1158
1159 float convfloat( unsigned char *string, int nbyte )
1160 {
1161     int          i,ii;

```

```

1162     float          numb=0.;
1163     long           lbig;
1164     unsigned char  *chrwrk,*cbig;
1165
1166     lbig=1;
1167     cbig = (unsigned char *)&lbig;
1168
1169     chrwrk = (unsigned char *)&numb;
1170     for( i = 0; i < nbyte; i++) {
1171         if( 1 == cbig[3] & 0x01 ) ii = 4 - nbyte + i;
1172         else                          ii = nbyte - 1 - i;
1173
1174         chrwrk[ii] = string[i];
1175     }
1176     return( numb );
1177 }
1178
1179 long convlatlon(unsigned char *string )
1180 {
1181     long    ll;
1182     char    c,a;
1183     unsigned char wk[5];
1184
1185     c = MASK1 & string[0];
1186     if( 0 != c ){
1187         a = -1;}
1188     else{
1189         a = 1;
1190     }
1191
1192     wk[0] = MASK7 & string[0];
1193     wk[1]=string[1];wk[2]=string[2];wk[3]=string[3];
1194     ll = (long)a*convlong(wk,(int)4);
1195     return(ll);
1196 }
1197
1198 /*=====
1199 */
1200 /*    used in identifydata                                */
1201 /*=====
1202 */
1203 void fproduct(void)
1204 {
1205     char    wk[4];
1206     fl[0]='X';
1207     fl[1]='X';
1208     fl[2]='X';
1209     if( mtn == 0)
1210     {
1211         if( pc == 0 && pn == 0 ){fl[0]='T';fl[1]='_';fl[2]='_';}
1212         if( pc == 0 && pn == 49){fl[0]='P';fl[1]='W';fl[2]='G';}
1213         if( pc == 0 && pn == 9 ){fl[0]='T';fl[1]='A';fl[2]='_';}
1214         if( pc == 1 && pn == 0 ){fl[0]='Q';fl[1]='Q';fl[2]='_';}

```

```

1215     if( pc == 1 && pn == 1 ){fl[0]='R';fl[1]='H';fl[2]='_';}
1216     if( pc == 1 && pn == 8 ){fl[0]='R';fl[1]='R';fl[2]='R';}
1217     if( pc == 1 && pn == 20 ){fl[0]='r';fl[1]='r';fl[2]='r';}
1218     if( pc == 1 && pn == 210 ){fl[0]='R';fl[1]='R';fl[2]='d';}
1219     if( pc == 1 && pn == 211 ){fl[0]='R';fl[1]='A';fl[2]='d';}
1220     if( pc == 1 && pn == 212 ){fl[0]='Q';fl[1]='Q';fl[2]='A';}
1221     if( pc == 1 && pn == 213 ){fl[0]='R';fl[1]='H';fl[2]='A';}
1222     if( pc == 2 && pn == 1 ){fl[0]='W';fl[1]='_';fl[2]='_';}
1223     if( pc == 2 && pn == 2 ){fl[0]='U';fl[1]='_';fl[2]='_';}
1224     if( pc == 2 && pn == 3 ){fl[0]='V';fl[1]='_';fl[2]='_';}
1225     if( pc == 2 && pn == 4 ){fl[0]='P';fl[1]='S';fl[2]='T';}
1226     if( pc == 2 && pn == 5 ){fl[0]='C';fl[1]='H';fl[2]='T';}
1227     if( pc == 2 && pn == 210 ){fl[0]='U';fl[1]='A';fl[2]='_';}
1228     if( pc == 2 && pn == 211 ){fl[0]='V';fl[1]='A';fl[2]='_';}
1229     if( pc == 3 && pn == 0 ){fl[0]='P';fl[1]='_';fl[2]='_';}
1230     if( pc == 3 && pn == 1 ){fl[0]='P';fl[1]='s';fl[2]='a';}
1231     if( pc == 3 && pn == 4 ){fl[0]='z';fl[1]='_';fl[2]='_';}
1232     if( pc == 3 && pn == 5 ){fl[0]='Z';fl[1]='_';fl[2]='_';}
1233     if( pc == 3 && pn == 8 ){fl[0]='P';fl[1]='A';fl[2]='_';}
1234     if( pc == 3 && pn == 9 ){fl[0]='Z';fl[1]='A';fl[2]='_';}
1235     }
1236     else if(mtn == 10 )
1237     {
1238         if( pc == 3 && pn == 0 ){fl[0]='S';fl[1]='T';fl[2]='_';}
1239         if( pc == 3 && pn == 192 ){fl[0]='S';fl[1]='T';fl[2]='A';}
1240     }
1241     fl[3]='?';
1242     fl[4]='?';
1243     fl[5]='?';
1244     if( nmem < 0 ){fl[3]='_';fl[4]='_';fl[5]='_';}
1245     if( nmem >= 0 && nmem < 100 ){
1246         switch (tyens)
1247         {
1248             case 1:
1249                 fl[5] = '_'; break;
1250             case 2:
1251                 fl[5] = 'm'; break;
1252             case 3:
1253                 fl[5] = 'p'; break;
1254             default:
1255                 printf(" tyens Not Supported !! tyens=%d ¥n",tyens);exit(71);
1256         }
1257         sprintf(wk,"%02d",nmem-1);fl[3]=wk[0];fl[4]=wk[1];
1258     }
1259     if( nmem >= 100 ){printf(" Over 100 members!! Not Supported.¥n");exit(70);}
1260 }
1261 }
1262
1263 void flevel(void)
1264 {
1265     char    wk[10];
1266     unsigned long vl1hpa;
1267     vl1hpa=vl1*0.01;

```

```

1268     fl[6]='X';
1269     fl[7]='X';
1270     fl[8]='X';
1271     if( tys == 1 ){fl[6]='S';fl[7]='F';fl[8]='C';}
1272     if( tys == 100 ){sprintf(wk,"%03d",v11hpa);fl[6]=wk[0];fl[7]=wk[1];fl[8]=wk[2];}
1273     if( tys == 101 ){fl[6]='S';fl[7]='E';fl[8]='A';}
1274     if( tys == 103 ){fl[6]='2';fl[7]='M';fl[8]='_';}
1275
1276     }
1277 void fderived(void)
1278     {
1279     fl[9]='X';fl[10]='X';fl[11]='X';
1280     if( dfn == 0 ){fl[9]='E';fl[10]='S';fl[11]='_';}/* Unweighted mean of all members */
1281     if( dfn == 1 ){fl[9]='E';fl[10]='S';fl[11]='W';}/* Weighted mean of all members */
1282     if( dfn == 4 ){fl[9]='S';fl[10]='P';fl[11]='R';}/* Spread of all members */
1283     if( dfn == 5 ){fl[9]='L';fl[10]='A';fl[11]='I';}/* Large Anomaly Index of all members */
1284     if( dfn == 6 ){fl[9]='C';fl[10]='L';fl[11]='S';}/* Unweighted mean of the cluster members */
1285
1286     if( Npdt == 1){fl[9]='M';fl[10]='E';fl[11]='M';}/* All member */
1287     }
1288 void fhemispher(void)
1289     {
1290     fl[12]='?';fl[13]='?';
1291     if(La1 >= 80000000 && La2 <= -80000000 ){fl[12]='-';fl[13]='G';}
1292     if(La1 == 90000000 && La2 == 0 ) {fl[12]='-';fl[13]='N';}
1293     if(La1 == 0 && La2 == -90000000 ){fl[12]='-';fl[13]='S';}
1294     if(La1 == 90000000 && La2 == -90000000 ){fl[12]='-';fl[13]='G';}
1295
1296     }
1297 void fperiod(void)
1298     {
1299     fl[14]='.';fl[15]='F';fl[16]='?';fl[17]='?';fl[18]='?';fl[19]='E';
1300     fl[20]='?';fl[21]='?';fl[22]='?';fl[23]='D';fl[24]='?';fl[25]='?';
1301     sprintf(&fl[14],".F%03dM%03dD%02d",kt,imm,iday);
1302
1303     }
1304
1305 /*=====
1306 */
1307 /* Identify Data (FILE NAME) */
1308 /*=====
1309 */
1310 void identifydata(void)
1311     {
1312     int i;
1313     for( i = 0 ; i < 30 ; ++i){
1314         fl[i]=' ';
1315     }
1316     fproduct();
1317     flevel();
1318     fderived();
1319     fhemispher();
1320     fperiod();

```

```
1321     fl[26]='.';fl[27]='d';fl[28]='a';fl[29]='t';fl[30]=0x00;  
1322     printf(" FNAME= %s¥n¥n",fl);  
1323     }  
1324  
1325
```

## 1か月予報アンサンプル統計格子点値ファイルの解読（デコード）処理について

1. 解読サンプルプログラムのソースコード  
別紙参照。
2. 利用方法  
以下、解読サンプルプログラムの  
ソースコードのファイル名：dcd\_1megrib1\_sample.c  
実行ファイル名                ：dcd\_1merib1  
です。
  - (1) ccコマンドによりコンパイルしてください。その際標準算術関数を利用可能なようにライブラリをリンクしてください。  
実行例) \$ cc dcd\_1megrib1\_sample.c -lm -o dcd\_1merib1  
(dcd\_1megrib2 という実行ファイルが生成されます)
    - ・ ANSI 準拠の c コンパイラでコンパイルできます。UNIX (HP-UX) 及び Linux (RedHat) での動作を確認しています。
    - ・ リトルエンディアンマシンにも対応しています。
  - (2) 次のコマンドを入力することにより、1か月予報アンサンプル統計格子点値、各節の内容が端末に表示されると共に、4バイト実数形式でデコードされたデータがファイルに書き出されます。  
実行例) \$ dcd\_1merib1 { 1か月予報アンサンプル統計格子点値ファイル名 }
    - ・ デコードされたデータは、予報時間ごと、要素ごとに、サンプルプログラムで割り付けた複数のファイルに出力されます。ファイル名は90行で使用されている関数out1\_fname495～520行で割り付けています。
    - ・ ファイルの出力を止めたい場合は、サンプルプログラム13行目の  
#define IFOUT 1  
を  
#define IFOUT 0  
など、1以外の数字に変更してください。

UNIX は、X/Open Company Limited が独占的にライセンスしている米国ならびに他の国における登録商標です。

Linux は、Linus Torvalds の米国及びその他の国における登録商標あるいは商標です。

HP-UX は、米国 Hewlett-Packard Company のオペレーティングシステムの名称です。

RedHat は、米国 Red Hat Software, Inc. の登録商標です。

**解読サンプルプログラムのソースコード**  
 ( 1 か月予報アンサンブル統計格子点値ファイル用 )

別紙

```

1  #include <stdio.h>
2  #include <string.h>
3  #include <math.h>
4  #include <stdlib.h>
5  #include <ctype.h>
6
7
8  #define SIZEOFBUF    20000
9  #define SIZEOFBITBUF 10000
10 #define MASK1       0x80
11 #define UNDEF       -19999.
12 #define MAX_IJX     20000
13 #define IFOUT       1      /* 1 :Output Data          */
14
15 /*=====
16 */
17 /*  DEFINITION OF GLOBAL PARAMETRES          */
18 /*=====
19 */
20 FILE          *fp,*fpc,*fpa,*fpg,*fp1w,*fp2w,*fpsh;
21 static unsigned char  buffer[SIZEOFBUF],map[SIZEOFBITBUF];
22 unsigned long        len;                      /* section 0  */
23 char                 infile[100];             /* section 0  */
24 unsigned long        iprm,ilvl,lvel;          /* section 1  */
25 unsigned long        iyy,imm,ihh,idd,imin,kt; /* section 1  */
26 long                 D,E,ifg,ifb;            /* section 1  */
27 unsigned long        Ni,Nj,Di,Dj,P1,P2;      /* section 2  */
28 long                 La1,Lo1,La2,Lo2;        /* section 2  */
29 static float         data[MAX_IJX];          /* section 4  */
30 /*=====
31 */
32 /* TYPE DEF          */
33 /*=====
34 */
35 /*-----*/
36 /*  TOOLS          */
37 /*-----*/
38 void usage(void);
39 long longbybit(unsigned char *buf,unsigned long *pos,unsigned long nbit);
40 long convlong( unsigned char *string, int nbyte );
41 float convfloat( unsigned char *string, int nbyte );
42 float m2ieee32(unsigned char *string);
43 long conv1latlon( unsigned char *string, int nbyte );
44 void element(char celem[]);
45 void out1_fname(void);
46 void out1_data(void);
47 void ini2date1(char ddmmyyyy[]);
48 /*-----*/

```

```

49  /*  DECODE SECTION                                     */
50  /*-----*/
51  void dcd1_sect0(void);
52  void dcd1_sect1(void);
53  void dcd1_sect2(void);
54  void dcd1_sect3(void);
55  void dcd1_sect4(void);
56  void dcd1_sect5(void);
57  /*=====
58  */
59  /* MAIN PROGRAM                                         */
60  /*-----
61  */
62  main(argc,argv)
63  int   argc;
64  char  **argv;
65  {
66      char          wk[105];
67      /*  Input File  */
68      if(2 != argc)
69          {
70              usage();
71              exit(0);
72          }
73      if(NULL == (fp = fopen(argv[1],"rb")))
74          {
75              printf("\nCannot open FILE : %s\n",argv[1]);
76              usage();
77              exit(1);
78          }
79
80      strcpy(infile,argv[1]);
81
82  SECT0:
83      dcd1_sect0();
84      dcd1_sect1();
85      if( ifg == 1 )dcd1_sect2();
86      if( ifb == 1 )dcd1_sect3();
87      dcd1_sect4();
88      dcd1_sect5();
89
90      out1_fname();
91      if( IFOUT == 1 ) out1_data();          /*  OutPut Data  */
92
93      goto SECT0;
94  }
95
96  /*-----*/
97  /*  SECTION 0: THE INDICATOR SECTION (IS)               */
98  /*-----*/
99  void dcd1_sect0(void)
100  {
101      unsigned long ngrib;

```

```

102
103     /* Check Header */
104     fread(buffer,sizeof(char), 8, fp);
105     if ( buffer[0] != 'G' || buffer[1] != 'R' ||
106         buffer[2] != 'T' || buffer[3] != 'B' ){
107         printf("Cannot Find GRIB header !!\n");
108         fclose(fp);
109         exit(0);
110     }
111     printf("-----\n");
112     printf("---- GRIB FILE !! ----\n");
113     printf("-----\n");
114     len    =convlong(&buffer[ 4], 3);
115     ngrib  =convlong(&buffer[ 7], 1);
116     printf("\n<<SECTION 0>> : Indicator Section\n");
117     printf("%10d : Total length, in octets, of GRIB messag\n",len);
118     printf("%10d : GRIB Edition number\n",ngrib);
119 }
120
121 /*-----*/
122 /* SECTION 1: THE PRODUCT DEFINITION SECTION (PDS) */
123 /*-----*/
124 void dcd1_sect1(void)
125 {
126     unsigned long  len1,ng,jma,np;
127     unsigned long  iunit,kds,kde,idx,nnn,nn2,icn,jma2,iend;
128     unsigned char  mask1=0x01;
129
130     if( fread(buffer,sizeof(char), 3, fp) == 0 ) exit(3);
131     len1  =convlong(&buffer[ 0], 3);
132     fseek(fp,-3,1);
133     if( fread(buffer,sizeof(char),len1, fp) == 0 ) exit(4);
134
135     ifg   = buffer[7] >> 7 & mask1 ;
136     ifb   = buffer[7] >> 6 & mask1 ;
137     iprm  =convlong(&buffer[ 8], 1);
138     ilvl  =convlong(&buffer[ 9], 1);
139     lvel  =convlong(&buffer[10], 2);
140     iyy   =convlong(&buffer[12], 1);
141     imm   =convlong(&buffer[13], 1);
142     idd   =convlong(&buffer[14], 1);
143     ihh   =convlong(&buffer[15], 1);
144     imin  =convlong(&buffer[16], 1);
145     kt    =convlong(&buffer[17], 1);
146     P1    =convlong(&buffer[18], 1);
147     P2    =convlong(&buffer[19], 1);
148     nnn   =convlong(&buffer[21], 2);
149     D     =convlong(&buffer[26], 2);
150     if(D > 32768) D = (D - 32768 )*(-1);
151     printf("\n<<SECTION 1>> : THE PRODUCT DEFINITION SECTION \n");
152     printf("%10d : Length in octets of the Product Definition Section\n",len1);
153     printf("%10d : Parameter Table Version numbe\n",buffer[3]);
154     printf("%10d : Identification of center (Table 0 - Part 1) \n",buffer[4]);

```

```

155     printf("%10d : Generating process ID number ¥n",buffer[5]);
156     printf("%10d : Grid Identification (geographical location and area; See Table B) ¥n",buffer[6]);
157     printf("%10d : Flag specifying the presence or absence of a GDS or a BMS ¥n",buffer[7]);
158     printf("    %10d : 0 GDS Omitted, 1 GDS Included¥n",ifg);
159     printf("    %10d : 0 BMS Omitted, 1 BMS Included¥n",ifb);
160     printf("%10d : Indicator of parameter and units(Table 2) ¥n",iprm);
161     printf("%10d : Indicator of type of level or layer(See Tables 3 & 3a) ¥n",ilvl);
162     printf("%10d : Height, pressure, etc. of the level or layer(See Table 3) ¥n",lvel);
163     printf("%10d : Year of century ¥n",iyy);
164     printf("%10d : Month of year ¥n",imm);
165     printf("%10d : Day of month ¥n",idd);
166     printf("%10d : Hour of day ¥n",ihh);
167     printf("%10d : Minute of hour ¥n",imin);
168     printf("%10d : Forecast time unit ¥n",kt);
169     printf("%10d : P1 - Period of time ¥n",P1);
170     printf("%10d : P2 - Period of time ¥n",P2);
171     printf("%10d : Time range indicator ¥n",buffer[20]);
172     printf("%10d : Number included in average¥n",nnn);
173     printf("%10d : Number Missing from averages or accumulations¥n",buffer[23]);
174     printf("%10d : Century of Initial (Reference) time (=20 until Jan. 1, 2001)¥n",buffer[24]);
175     printf("%10d : Identification of sub-center ¥n",buffer[25]);
176     printf("%10d : The decimal scale factor D. ¥n",D);
177     printf("¥n");
178 }
179
180 /*-----*/
181 /* SECTION 2: GRID DESCRIPTION SECTION (GDS) */
182 /*-----*/
183 void dcd1_sect2(void)
184 {
185     unsigned long len2,nv,ipv,ityp;
186     unsigned long ifr,iflg;
187
188     if( fread(buffer,sizeof(char), 3, fp) == 0 ) exit(5);
189     len2  =convlong(&buffer[ 0], 3);
190     fseek(fp,-3,1);
191     if( fread(buffer,sizeof(char),len2, fp) == 0 ) exit(6);
192
193     nv    =convlong(&buffer[ 3], 1);
194     ipv   =convlong(&buffer[ 4], 1);
195     ityp  =convlong(&buffer[ 5], 1);
196
197     printf("¥n<<SECTION 2>> : GRID DESCRIPTION SECTION (GDS) ¥n");
198     printf("%10d : Length in octets of the Grid Description Section ¥n",len2);
199     printf("%10d : NV, the number of vertical coordinate parameters,255: ¥n",nv);
200     printf("%10d : PV or PL or 255: =255:neither are present¥n",ipv);
201     printf("%10d : Data representation type (See Table 6) ¥n",ityp);
202     switch(ityp){
203     case 0:
204         Ni  =convlong(&buffer[ 6], 2);
205         Nj  =convlong(&buffer[ 8], 2);
206         La1 =conv1latlon(&buffer[10], 3);
207         Lo1 =convlong(&buffer[13], 3);

```

```

208     ifr  =convlong(&buffer[16], 1);
209     La2  =conv1latlon(&buffer[17], 3);
210     Lo2  =convlong(&buffer[20], 3);
211     Di   =convlong(&buffer[23], 2);
212     Dj   =convlong(&buffer[25], 2);
213     ifg  =convlong(&buffer[27], 1);
214
215     printf("¥n          << LATITUDE/LONGITUDE GRIDS INCLUDING GAUSSIAN >> ¥n");
216     printf("%10d : Ni - No. of points along a latitude circle¥n",Ni);
217     printf("%10d : Nj - No. of points along a longitude meridian¥n",Nj);
218     printf("%10d : La1 - latitude of first grid point units:(degrees x 1000) ¥n",La1);
219     printf("%10d : Lo1 - longitude of first grid point units:(degrees x 1000) ¥n",Lo1);
220     printf("%10d : Resolution and component flags (Table 7)¥n",ifr);
221     printf("%10d : La2 - Latitude of last grid point¥n",La2);
222     printf("%10d : Lo2 - Longitude of last grid point¥n",Lo2);
223     printf("%10d : Di - Longitudinal Direction Increment ¥n",Di);
224     printf("%10d : Dj - Latitudinal Direction Increment ¥n",Dj);
225     break;
226     default:
227         printf("Unsupported GDS !!");
228         exit(20);
229     }
230 }
231
232 /*-----*/
233 /* SECTION 3: BIT MAP SECTION (BMS) */
234 /*-----*/
235 void dcd1_sect3(void)
236 {
237     unsigned long len3,lodd,nu;
238
239     if( fread(buffer,sizeof(char), 6, fp) == 0 ) exit(7);
240     len3  =convlong(&buffer[ 0], 3);
241     lodd  =convlong(&buffer[ 3], 1);
242     nu    =convlong(&buffer[ 4], 2);
243     if( fread(map,sizeof(char),len3-6, fp) == 0 ) exit(8);
244
245     printf("¥n<<SECTION 3>> : BIT MAP SECTION (BMS) ¥n");
246     printf("%10d : Length in octets of Bit Map Section ¥n",len3);
247     printf("%10d : Number of unused bits at end of Section 3¥n",lodd);
248     printf("%10d : Numeric: = 0: a bit map follows;otherwise..... ¥n",nu);
249 }
250
251 /*-----*/
252 /* SECTION 4: BINARY DATA SECTION (BDS) */
253 /*-----*/
254 void dcd1_sect4(void)
255 {
256     float          R;
257     float          pow2E,pow10D;
258     unsigned char  msk_bit=MASK1;
259     unsigned long  posi,lgpv,i,j,nb,mb,i_out;
260     unsigned char  cwk[10];

```

```

261     unsigned long len4,iflg,lodd4,nbit;
262
263     if( fread(buffer,sizeof(char),11, fp) == 0 ) exit(9);
264     len4   =convlong(&buffer[ 0], 3);
265
266     cwk[0] =  buffer[ 3] >> 4 & 0x0f;
267     iflg   =convlong(&cwk[0], 1);
268
269     cwk[0] =  buffer[ 3] & 0x0f;
270     lodd4  =convlong(&cwk[0], 1);
271
272     cwk[0]=buffer[ 4] & 0x7f;
273     cwk[1]=buffer[ 5];
274     E      = convlong(&cwk[0],2);
275     if ( buffer[ 4] >> 7 & 0x01 == 1 ) E= -1*E;
276
277     R      =m2ieee32(&buffer[6]);          /* M -> ieee32 */
278     nbit   =convlong(&buffer[10], 1);
279
280     printf("¥n<<SECTION 4>> : BINARY DATA SECTION (BDS) ¥n");
281     printf("%10d : Length in octets of binary data section¥n",len4);
282     printf("    %10d : Bits 1 through 4: Flag - See Table 11 ¥n",iflg);
283     printf("    %10d : Bits 5 through 8: Number of unused bits at end of Section 4¥n",lodd4);
284     printf("%10d : The binary scale factor (E) ¥n",E);
285     printf("%10.4f : Reference value (minimum value) ¥n",R);
286     printf("%10d : Number of bits into which a datum point is packed ¥n",nbit);
287
288     if( fread(buffer,sizeof(char),len4-11, fp) == 0 ) exit(10);
289         if (E >= 0) pow2E=pow(2.0,(double)E);
290         else      pow2E=1.0/pow(2.0,-1.0*(double)E);
291         if (D >= 0) pow10D=pow(10.0,(double)D);
292         else      pow10D=1.0/pow(10.0,-1.0*(double)D);
293         posi = 0;
294         for( i = 0 ;i < Ni*Nj ;i++){
295             nb = i / 8;
296             mb = i % 8;
297             if( 0 == (map[nb] & ( msk_bit >> mb )) && ifb == 1 ){
298                 data[i]=UNDEF;
299             }
300             else{
301                 lgpv=longbybit(buffer,&posi,nbit);
302                 /*data[i] = (R + lgpv*pow(2.0,(double)E))/pow(10.0,(double)D);*/
303                 data[i] = (R + lgpv*pow2E)/pow10D;
304             }
305         }
306         /*data[2933]=0.0;data[3513]=0.0;*/
307         i_out=50;
308         printf("Check DATA !!    ¥n");
309         printf("    No    value (i_out=%d)    ¥n",i_out);
310         for(j = 0; j < Nj ; ++j){
311             printf("    %5d    %f¥n",j+1,data[j*Ni+i_out]);
312         }
313

```

```

314 }
315
316 /*-----*/
317 /* SECTION 5: END SECTION */
318 /*-----*/
319 void dcd1_sect5(void)
320 {
321     if( fread(buffer,sizeof(char), 4, fp) == 0 ) exit(9);
322     if(buffer[0] == '7' && buffer[1] == '7' && buffer[2] == '7' && buffer[3] == '7')
323         printf("GRIB1 END !!\n");
324         printf("\n\n");
325 }
326
327 /*=====*/
328 /* TOOLS */
329 /*=====*/
330
331 void usage(void)
332 {
333     puts("\n");
334     puts("Usage : GRIB1 <FILE-NAME>");
335 }
336
337 long longbybit(unsigned char *buf,unsigned long *pos,unsigned long nbit)
338 {
339     char    i,ii;
340     long    pos8,mg8,rem8,nbytes,remnbit;
341     long    lout=0;
342     long    lbig;
343     unsigned char    *buf2,*cbig;
344     unsigned char    mask,maskremn,full=0xff;
345
346     lbig=1;
347     cbig = (unsigned char *)&lbig
348
349     buf2 = (unsigned char *)&lout
350     mg8 = *pos % 8;
351     remnbit= nbit % 8;
352
353     if( (*pos + nbit) % 8 == 0 ) rem8 = 0;
354     else rem8 = 8 - (( *pos + nbit ) % 8 );
355
356     pos8 = ( *pos + nbit + rem8 ) / 8 - 1;
357
358     nbytes = ( nbit ) / 8;
359     mask = full >> mg8 + rem8;
360     for( i = 0;i < nbytes ;i++){
361         if( 1 == cbig[3] & 0x01 ) ii = 3 -i;
362         else ii = i ;
363         buf2[ii] = ( buf[pos8 - i] >> rem8
364             | buf[pos8 - i - 1] << 8 - rem8 );
365     }
366

```

```

367     if( 1 == cbig[3] & 0x01 ) ii = 3 - nbytes;
368     else                               ii = nbytes ;
369     if( remnbit <= 8 - rem8 ){
370         maskremn = full >> 8 - remnbit;
371         buf2[ii] = ( buf[pos8 - nbytes] >> rem8 & maskremn );
372     }
373     else{
374         maskremn = full >> 8 - remnbit & full << 8 - rem8 ;
375         buf2[ii] = ( buf[pos8 - nbytes] >> rem8
376             | buf[pos8 - nbytes - 1] << 8 - rem8 & maskremn );
377     }
378     *pos = *pos + nbit;
379     return(lout);
380 }
381
382 long convlong( unsigned char *string, int nbyte )
383 {
384     int          i,ii;
385     long         numb=0;
386     long         lbig;
387     unsigned char *chrwrk,*cbig;
388
389     lbig=1;
390     cbig = (unsigned char *)&lbig;
391     /*if( 1 == cbig[3] & 0x01 ) printf("BIG ENDIAN !!");*/
392     chrwrk = (unsigned char *)&numb;
393     for( i = 0; i < nbyte; i++ ) {
394         if( 1 == cbig[3] & 0x01 ) ii = 4 - nbyte + i;
395         else                               ii = nbyte - 1 - i;
396
397         chrwrk[ii] = string[i];
398     }
399     return( numb );
400 }
401
402 float m2ieee32(unsigned char *string)
403 {
404     float      R,pow16iax;
405     unsigned   long ia,ib;
406     long       iax,is;
407     unsigned char cwk[10];
408
409     ib      =convlong(&string[1], 3);
410     cwk[0]= string[0] & 0x7f ;
411     ia      =convlong(&cwk[0],1);
412
413     if ( string[0] >> 7 & 0x01 == 1)is=-1;
414     else                               is= 1;
415
416     iax=ia-64;
417     if( iax >= 0) pow16iax=pow(16,(double)iax);
418     else          pow16iax=1.0/pow(16,(double)iax);
419

```

```

420     R      =(double)is*(double)ib*pow16iax/pow(2.0,24);
421     return(R);
422 }
423
424 float convfloat( unsigned char *string, int nbyte )
425 {
426     int          i,ii;
427     float        numb=0.;
428     long         lbig;
429     unsigned char *chrwrk,*cbig;
430
431     lbig=1;
432     cbig = (unsigned char *)&lbig
433
434     chrwrk = (unsigned char *)&numb
435     for( i = 0; i < nbyte; i++ ) {
436         if( 1 == cbig[3] & 0x01 ) ii = 4 - nbyte + i;
437         else                          ii = nbyte - 1 - i;
438
439         chrwrk[ii] = string[i];
440     }
441     return( numb );
442 }
443
444 long conv1latlon( unsigned char *string, int nbyte )
445 {
446     int          i,ii;
447     long         numb=0;
448     long         lbig;
449     unsigned char *chrwrk,*cbig;
450     unsigned char cwk;
451
452     lbig=1;
453     cbig = (unsigned char *)&lbig
454     /*if( 1 == cbig[3] & 0x01 ) printf("BIG ENDIAN !!");*/
455     chrwrk = (unsigned char *)&numb
456
457     for( i = 0; i < nbyte; i++ ) {
458         if( i == 0 ) cwk= string[i] & 0x7f;
459         else          cwk= string[i];
460
461         if( 1 == cbig[3] & 0x01 ) ii = 4 - nbyte + i;
462         else                          ii = nbyte - 1 - i;
463         chrwrk[ii] = cwk;
464     }
465     if ( 1 == 0x01 & string[0] >> 7 )numb= -1*numb;
466     return( numb );
467 }
468
469 void element(char celem[])
470 {
471     if (iprm == 7 && ilvl == 100 && lvel == 500){ strcpy(celem, "Z500") ;}
472     else if(iprm == 27 && ilvl == 100 && lvel == 500){ strcpy(celem, "Z500ANOM") ;}

```

```

473     else if(iprm == 142 && ilvl == 100 && lvel == 500){ strcpy(celem, "Z500SPRD") ;}
474     else if(iprm == 140 && ilvl == 100 && lvel == 500){ strcpy(celem, "Z500HANM") ;}
475     else if(iprm == 11 && ilvl == 100 && lvel == 850){ strcpy(celem, "T850") ;}
476     else if(iprm == 25 && ilvl == 100 && lvel == 850){ strcpy(celem, "T850ANOM") ;}
477     else if(iprm == 143 && ilvl == 100 && lvel == 850){ strcpy(celem, "T850SPRD") ;}
478     else if(iprm == 2 && ilvl == 102 && lvel == 0){ strcpy(celem, "PSEA") ;}
479     else if(iprm == 141 && ilvl == 102 && lvel == 0){ strcpy(celem, "PSEASPRD") ;}
480     else if(iprm == 26 && ilvl == 102 && lvel == 0){ strcpy(celem, "PSEAANOM") ;}
481     else if(iprm == 61 && ilvl == 1 && lvel == 0){ strcpy(celem, "RAIN") ;}
482     else if(iprm == 52 && ilvl == 100 && lvel == 850){ strcpy(celem, "H850") ;}
483     else if(iprm == 33 && ilvl == 100 && lvel == 850){ strcpy(celem, "U850") ;}
484     else if(iprm == 34 && ilvl == 100 && lvel == 850){ strcpy(celem, "V850") ;}
485     else if(iprm == 33 && ilvl == 100 && lvel == 200){ strcpy(celem, "U200") ;}
486     else if(iprm == 34 && ilvl == 100 && lvel == 200){ strcpy(celem, "V200") ;}
487     else if(iprm == 7 && ilvl == 100 && lvel == 100){ strcpy(celem, "Z100") ;}
488     else if(iprm == 27 && ilvl == 100 && lvel == 100){ strcpy(celem, "Z100ANOM") ;}
489     else {
490         printf("This element is not supported !! iprm ilvl vlevel %d %d %d¥n",iprm,ilvl,lvel);
491         exit(30);}
492
493 }
494
495 void out1_fname()
496 {
497     char fname[100],fnamec[100];
498     char celem[100];
499     char chemisp[5];
500     char cperiod[10];
501     char ddmmyyyy[20];
502     float flo1,fdi,fla2,fdj;
503     /* element */
504     element(celem);
505     /* hemisphere */
506     if (La1 > 0 && La2 >= 0){strcpy(chemisp,"N");}
507     else if (La1 <= 0 && La2 < 0){strcpy(chemisp,"S");}
508     else if (La1 > 0 && La2 < 0){strcpy(chemisp,"G");}
509     else { printf("Not supported hemispher ??¥n");
510           exit(30);}
511
512     /* period */
513     sprintf(cperiod,"_D%02d%02d",P1,P2);
514
515     sprintf(fname, "%s%s%s",celem,chemisp,cperiod);
516     sprintf(fnamec,"%s%s%s.s.ctl",celem,chemisp,cperiod);
517
518     /*printf("fname =%s ¥nfnamec=%s¥n",fname,fnamec);*/
519     printf("FNAME=%s¥n",fname);
520 }
521
522
523 void out1_data()
524 {
525     char fname[100],fnamec[100];

```

```

526     char celem[100];
527     char chemisp[5];
528     char cperiod[10];
529     char ddmmyyyy[20];
530     float flo1,fdi,fla2,fdj;
531     /* element */
532     element(celem);
533     /* hemisphere */
534     if ( La1 > 0 && La2 >= 0 ){strcpy(chemisp,"N");}
535     else if ( La1 <= 0 && La2 < 0 ){strcpy(chemisp,"S");}
536     else if ( La1 > 0 && La2 < 0 ){strcpy(chemisp,"G");}
537     else { printf("Not supported hemispher ??¥n");
538           exit(30);}
539
540     /* period */
541     sprintf(cperiod,"_D%02d%02d",P1,P2);
542
543     sprintf(fname ,"%s%s%s",celem,chemisp,cperiod);
544     sprintf(fnamec,"%s%s%s.sctl",celem,chemisp,cperiod);
545
546     /*printf("fname =%s ¥nfnamec=%s¥n",fname,fnamec);*/
547     /*printf("FNAME=%s¥n",fname);*/
548
549     /* OutPut DATA (GrDAS Format) */
550     if(( fpg = fopen(fname,"wb")) == NULL){
551         printf("Stop, Create File(%s)¥n",fname);
552         exit(97);
553     }
554     if( fwrite(data,4,Ni*Nj,fpg) != Ni*Nj ) {
555         printf("Write ERROR !! File(%s)¥n",fname);
556         exit(98);
557     }
558     if ( fclose(fpg) != 0 ){
559         printf("Close ERR !! File(%s)¥n",fname);
560         exit(95);}
561
562     /* GrADS Control File */
563     if(( fpc = fopen(fnamec,"wt")) == NULL){
564         printf("Stop, Create File(%s)¥n",fnamec);
565         exit(99);
566     }
567     flo1=(float)Lo1/1000.0;
568     fla2=(float)La2/1000.0;
569     fdi=(float)Di/1000.0;
570     fdj=(float)Dj/1000.0;
571
572     ini2date1(ddmmyyyy);
573
574     fprintf(fpc,"dset ^%s¥n",fname);
575     fprintf(fpc,"undef -1999.¥n");
576     fprintf(fpc,"xdef %d linear %f %f¥n",Ni,flo1,fdi);
577     fprintf(fpc,"ydef %d linear %f %f¥n",Nj,fla2,fdj);
578     fprintf(fpc,"zdef 1 linear 0 1 ¥n");

```

```

579     fprintf(fpc,"tdef 1 linear %s 7dy¥n",ddmmyyyy);
580     fprintf(fpc,"vars 1 ¥n");
581     fprintf(fpc,"%s 0 99 Comment !!¥n",celem);
582     fprintf(fpc,"endvars¥n");
583     fprintf(fpc,"options yrev¥n");
584
585     if ( fclose(fpc) != 0 ){
586         printf("Close ERR !! File(%s)¥n",fnamec);
587         exit(96);}
588     else{
589         printf("Control File Closed !! = %s ¥n",fnamec);
590     }
591 }
592
593 void ini2date1(char ddmmyyyy[])
594 {
595     unsigned long yyyy1,mm1;
596     char yyyy[5],mm[3];
597
598     if ( iyy >= 70 )yyyy1=1900 + iyy;
599     else          yyyy1=2000 + iyy;
600
601     if ( imm == 1){ strcpy(mm,"Jan");}
602     else if( imm == 2){ strcpy(mm,"Feb");}
603     else if( imm == 3){ strcpy(mm,"Mar");}
604     else if( imm == 4){ strcpy(mm,"Apr");}
605     else if( imm == 5){ strcpy(mm,"May");}
606     else if( imm == 6){ strcpy(mm,"Jun");}
607     else if( imm == 7){ strcpy(mm,"Jul");}
608     else if( imm == 8){ strcpy(mm,"Aug");}
609     else if( imm == 9){ strcpy(mm,"Sep");}
610     else if( imm == 10){ strcpy(mm,"Oct");}
611     else if( imm == 11){ strcpy(mm,"Nov");}
612     else if( imm == 12){ strcpy(mm,"Dec");}
613     else{ printf(" Initial month is strange!! imm=%d¥n",imm); exit(55);}
614
615     sprintf(ddmmyyyy,"%02d%s%04d",idd,mm,yyyy1);
616 }
617
618

```