A DISCUSSION OF GRIB2 AND THE CAPABILITIES OF RELATED MDL SOFTWARE

Harry R. Glahn
Bryon Lawrence

1. Introduction

GRIB is the current World Meteorological Organization (WMO) standard for exchange of GRIdded Binary data. A number of shortcomings of GRIB have been identified, and GRIB2 has been developed.

The original GRIB provided for "Simple" and "Complex" packing. Simple packing was very straightforward and consisted essentially of only removing the smallest value from all the data values in the data field and then "packing" each value into the smallest number of bits necessary to contain the largest of the remaining values. The possibility of a bit map was provided for, in which case missing values would not be packed, but would be indicated by a “0" in a bit map otherwise consisting of “1's”.

Complex was somewhat more complex, but only somewhat. After subtracting the overall minimum of the data field, the minimum of each of several groups (say there are NG of them) was also removed from the corresponding group, and the remaining data values were packed. This necessitated putting into the packed GRIB "message" those NG minimum values and an indication of where the groups started and stopped, the latter being achieved with a "secondary" bit map. The method of splitting into groups was not specified, but a special case of row-by-row was accommodated. Row-by-row splitting was not generally very efficient; the method of determining groups was all-important in determining how well Complex packing worked.

In GRIB2, the Simple method is left basically intact. Missing values can be indicated by a bit map. However, the Complex method has been dramatically changed, and except that it still has the concept of groups bears little resemblance to the original Complex method. In addition, a third packing method called "Spatial Differencing” has been added. In Complex and Spatial Differencing, the method of defining groups, as it was with the original GRIB, is very important, and the efficiency of these packing methods depends on finding groups in which the data values are similar; this is aided by a boustrophedonic ordering of points (this does no good in the Simple method). In addition, a more efficient way of specifying missing values is incorporated, as well as the provision for two "kinds" of missing values. Each "section" in a GRIB2 message has a basic definition that applies to the whole section, then there can be one or more "Templates" further describing the data or the way the values are represented.

The Meteorological Development Laboratory (MDL) has written Fortran software to implement certain options of GRIB2. The packer is called **PK_GRIB2** and the unpacker is called **UNPK_GRIB2**. This software is in the process of being hardened, and is being provided to the user community as part of the National Weather Service's effort to have standardized encoders and decoders for various data forms.

Many values are required to be furnished to PK_GRIB2 to pack all the sections of GRIB2. Generally, these data are furnished to PK_GRIB2, and returned by UNPK_GRIB2, in arrays corresponding to the section numbers; e.g., an array IS1( )

will contain data for Section 1, where the location in the array corresponds to the starting octet for the octets in the section (e.g., IS5(10) will specify which template will be used in Section 5).

The following sections describe the options in GRIB2 currently supported by MDL software as well as MDL's interpretation of Sections 5, 6, and 7. In a few cases, elements in those sections are further defined. They also indicate the options in those sections supported by MDL software; portions of footnotes dealing only with MDL software and not the code form itself are in brackets. It is hoped this arrangement of definitions will help the user in understanding the details of the code form.

As indicated in Section 3, not all options (Templates, Code Tables, Flag Tables) are supported by MDL software. The definitions in the tables generally pertain to rectangular grids of size NX by NY - extensions to other types of grids may or may not be applicable.

In the following tables, the values in the octets are integer unless otherwise specified, except for the data values themselves in Section 7, which are not byte-oriented.

GRIB2 Local Use Section 2

MDL software provides for integer and floating point data in the Local Use Section. Documentation found in the PK_GRIB2 and UNPK_GRIB2 routines explains how such data are treated. To conserve on space, the local use data elements are packed into Section 2 of the GRIB2 message using the simple packing method.

GRIB2 SECTION 3

The following Section 3 templates are supported by the MDL GRIB2 encoder/decoder.

Template 3.0 (Latitude/Longitude)
Template 3.10 (Mercator)
Template 3.20 (Polar Stereographic Projection)
Template 3.30 (Lambert Conformal)
Template 3.90 (Space View Perspective or Orthographic)
Template 3.110 (Equatorial Azimuthal Equidistant Projection)
Template 3.120 (Azimuth-range Projection)

GRIB2 SECTION 4

The following Section 4 templates are supported by the MDL GRIB2 encoder/decoder.

Template 4.0 (Analysis or Forecast at a Horizontal Level or in a Horizontal Layer at a Point in Time)
Template 4.1 (Individual Ensemble Forecast, Control and Perturbed, at a Horizontal Level or in a Horizontal Layer at a Point in Time)
Template 4.2 (Derived Forecast Based on All Ensemble Members at a Horizontal Level or in a Horizontal Layer at a Point in Time)
Template 4.8 (Average, Accumulation, and/or Extreme Values at a Horizontal Level or in a Horizontal Layer in a Continuous or Non-continuous Time Interval)
Template 4.20 (Radar Product)

Template 4.30 (Satellite Product)

2. GRIB2 Section 5

The following table contains the descriptions of the elements of Section 5 by octet (byte) number. Footnotes are provided in an attempt to give a full description of the table entries. Note that text related only to MDL software is in brackets.

Contents of Section 5. When "same" appears in a box, it means the same as the box to the left. An asterisk on octet numbers means this has to be furnished by the user; a double asterisk means it has to be furnished, but may be changed by PK_GRIB2; other values are always filled by PK_GRIB2. The full array IS0( ) is always returned by UNPK_GRIB2 as it is found in the GRIB2 message.

Octets	Definition		
	Simple	Complex	Spatial Differencing
1-4	Length of section in bytes	Same	Same
5	Number of section = 5	Same	Same
6-9	Actual number of data points (NP) in Section 7[1]	Same[2]	Same[3]

[1]This is the number of non-missing values, which will be NX*NY if there are no missing values; therefore, no bit map is necessary. If there is a bit map, then it is the number of 1's in the bit map - the number of non-missing values. It is possible a bit map of all 1's could be present, but would be abnormal. A bit map of all 0's would occur if a field were packed but all values were missing. [The input to PK_GRIB2 can include (1) a bit map, in which case the "missing" values have already been removed and will not be packed, or (2) the missing values are still in the grid, a bitmap will be generated, and the corresponding values will be removed from the grid before packing. A bit map is returned by UNPK_GRIB2 if a bit map was packed into Section 6 of the GRIB2 message. If this is the case, then the user is given the option of having the data field returned to him with or without the missing values embedded within it.

[2]Footnote 1 applies when a bit map is used. If a bit map is not used, then this is always NX*NY points. The number of values actually packed may be less than this when a group of all the same value exists, and therefore, only the group reference is packed. This value is actually superfluous and is not needed for unpacking. [A bit map is not supported by the MDL PK_GRIB2 routine for the Complex or Spatial Differencing packing methods; rather, missing values are carried another way, and there is a "value" for every gridpoint. If the user does supply a bit map along with a data field that does not have any missing values in it, and the intent is to use the Complex or Spatial Differencing method to pack the data, PK_GRIB2 will insert the missing values into the data field and then pack the data. However, the UNPK_GRIB2 routine will recognize a bit map packed into Section 6 of a GRIB2 message for the Complex method. In this case, the caller of UNPK_GRIB2 will be given the option of receiving the data field either stripped of all of the missing values or with the missing values embedded

Octets	Definition		
	Simple	Complex	Spatial Differencing
10-11**	Template No. = 0	Template No. = 2	Template No. = 3
12-15	Reference value (R) in floating point[4]	Same	Same
16-17*	Binary scaling (E)	Same	Same
18-19*	Decimal scaling (D)	Same	Same
20	Number of bits used for each data value[5]	Number of bits used for each group reference value (the group minimum values)	Same
21*	Type of original field: 0 = floating point.[6] 1 = integer[7]	Same	Same
22*	-----	Basic splitting method:	Same

within it. In the case where the missing values are removed from the data field, the user will be given a bit map along with the value of the primary missing value (MISSP or XMISSP). Note that a bit map can only accommodate a primary missing value, not a primary and a secondary.]

[3]This is NX*NY values, the number of second-order differences plus 2. This value is actually superfluous and is not needed by unpacking software. The first two "non-missing" values are dummy, and are not used. See Footnote 2 for special notes on PK_GRIB2, Spatial Differenencing, and the use of a bit map. [Note that UNPK_GRIB2 does not recognize a bit map supplied in a GRIB2 message whose data has been packed using Spatial Differencing.]

[4]R is the minimum value in the field after multiplying the values by 10^D.

[5]If all (non-missing) values are equal, only the reference (octets 12-15) is needed and the data values are packed with zero bits per point. The number of values in octets 6-9 is not affected.

[6][The grid of values is provided to PK_GRIB2 in a REAL array and is returned to the user in a REAL array by UNPK_GRIB2.]

[7][The grid of values is provided to PK_GRIB2 in an INTEGER array and is returned to the user by UNPK_GRIB2 in an INTEGER array.]

Octets	Definition		
	Simple	Complex	Spatial Differencing
		0 = row-by-row[8] 1 = General[9]	
23**[10]	-----	Provision for missing values[11]: 0 = No explicit missing values included in the grid[12] 1 = Primary missing values included in the grid[13]	Same

[8][Not supported by MDL software.]

[9]This is a method whereby groups of non-uniform size are formed according to the pattern of consecutive data values. The specific algorithm for determining groups is not specified. [Note that row by row splitting is not supported by MDL software.] Octets 22 and above are not present for Simple Packing.

[10]Inclusion in GRIB2 of this value allows any value to be used as primary and secondary missing values, [MISSP (or XMISSP) and MISSS (or XMISSS),] respectively, even zero. Note that secondary missing values are not accommodated with a bit map; [A bit map is not packed into a GRIB2 message by the MDL PK_GRIB2 routine for the Complex and Spatial Differencing methods. If the user supplies a bit map with a data field that has no missing values in it, and the user intends to pack the data using the Complex or Differencing methods, then the bit map will be used to place the missing values into the data field before it is packed. A bit map is supported by the MDL UNPK_GRIB2 routine for the Complex method].

[11]For the Complex and Spatial Differencing methods, primary missing values are carried as the largest value possible in a particular group (all 1's in the field width (see octet 37). Secondary missing values are carried as the largest value possible in a particular group minus 1 [a zero for a missing value in the least significant bit with all preceding bits = 1 (see octet 37)].

[12]This would be used when there are no missing values or when a bit map is used. Note that this pertains to the packed message, not the data supplied to PK_GRIB2. [A bit map is not packed into a GRIB2 message by the MDL PK_GRIB2 routine for the Complex and Spatial Differencing methods. If the user supplies a bit map with a data field that has no missing values in it, and the user intends to pack the data using the Complex or Differencing methods, then the bit map will be used to place the missing values into the data field before it is packed. A bit map is supported by the MDL UNPK_GRIB2 routine for the Complex method.]

[13]Primary missing values are carried as the largest value possible in a

Octets	Definition		
	Simple	Complex	Spatial Differencing
		2 = Primary and secondary missing values included in the grid[14]	
24-27	-----	Primary missing value[15]	Same
28-31	-----	Secondary missing value[16]	Same
32-35	-----	Number of groups into which the data are split (NG)	Same

particular group (all 1's in the field width (see octet 37)). Not used with a bit map. The field width of the group has to be increased by 1 bit when the largest value that will fit in the original field width for that group is a non-missing value. For instance, if primary missing values were possible but not secondary missing values, and if the largest value in a group were 6, a missing value could be carried as a 7 and the field width of 3 bits would not have to be increased. However, if the largest value in a group were 7, then 15 in a 4-bit group field width would be reserved for the missing value.

[14]Secondary missing values are carried as the largest value possible in a particular group minus 1--a zero in the least significant bit with all other bits = 1 (see octet 37). Not used with a bit map. The field width for a group has to be increased by 1 bit when the second largest value that will fit in the original field width for that group is a non-missing value. If all of the values in a group are the same (which in the absence of missing values would mean that the group could be packed with a field width of 0), then the field width will be 2 if secondary missing values are present. Note that secondary missing values cannot be present unless primary values can also be present. This does not mean that primary values have to be present, but only that they could be. For instance, if both primary and secondary missing values were possible, and if the largest value in a group were 6, a missing value of 15 (1111 binary) in a 4-bit group field width would be reserved for the primary missing value and a 14 (1110 binary) would be reserved for the secondary missing value.

[15]This value is floating point or integer depending on octet 21. [The value is input to PK_GRIB2 and is output by UNPK_GRIB2.]

[16]This value is floating point or integer depending on octet 21. [The value is input to PK_GRIB2 and is output by UNPK_GRIB2.]

Octets	Definition		
	Simple	Complex	Spatial Differencing
36	-----	Reference value for the widths of NG groups (see octet 37)[17]	Same
37	-----	Number of bits used to specify the widths of NG groups, after the reference (octet 36) is removed[18]	Same
38-41	-----	Reference value for the lengths of NG groups (see octet 42)[19]	Same
42	-----	Length increment for the group lengths[20]	Same

[17]The "width" of a group is the number of bits necessary to hold the largest value in the group after the group reference is removed. This sometimes has to be increased by 1 or 2 to accommodate primary missing and secondary missing values (see footnotes 13 and 14 above).

[18]After splitting into groups, PK_GRIB2 determines the sizes of the groups and the number of bits necessary to hold the largest of those sizes. This is done after the smallest size is subtracted, thereby sometimes reducing the number of bits required; this smallest size is the "reference value" in octet 36.

[19]The "length" of a group is the number of values it contains. Once the lengths of all of the groups have been determined, the PK_GRIB2 routine determines the minimum of these lengths. This minimum length is the reference length that is stored in octets 38-41. It is subtracted from each of the group lengths, ultimately reducing the amount of storage space required to store these group lengths in the GRIB2 message.

[20]Some group splitting algorithms may employ logic that creates groups whose lengths are a common multiple of some base increment factor. For example, an algorithm may create groups whose lengths are a multiple of 5. So the possible lengths of groups produced by such an algorithm would be 5,10,15,20,25,30,35,40. In this example, the length increment for the group lengths would be 5. [The MDL group splitting algorithm does not work this way. The length increment is always set to 1.]

Octets	Definition		
	Simple	Complex	Spatial Differencing
43-46	-----	True Length of last group[21]	Same
47	-----	Number of bits used to specify lengths of NG groups, after reference (octets 38-41) is removed[22]	Same
48	-----	-----	Order of spatial differencing[23]: 0 = Not used 1 = First order differencing[24] 2 = second order differencing[25]

[21]Using the group splitting scheme outlined above in footnote 20, often the last group will have a length that cannot be an whole multiple of the increment factor. This would occur, for example, if the user is packing a field containing 27 data values with a length increment of 5. Suppose the group splitting method produces groups with lengths of 5, 10, 5, and 5. The last group must have a length of 2. Since this length is not a multiple of of the length increment, it is stored in octets 43-46 without subtracting the length reference from it and without scaling it.

[22]After splitting into groups, the lengths of the groups (i.e. the numbers of values in each group) are known. The number of bits necessary to hold the largest of those lengths is calculated after the reference length is subtracted from each group length and the resulting difference is divided by the length increment. This sometimes will reduce the number of bits required; the reference length is the "reference value" in octets 38-41 while the length increment is the value specified in octet 42. The length of a group can be up to the number of values in the grid, so can be quite large. Large groups can happen when a field is mostly of one value, or composed of largely missing values. Although it would be unusual for the reference value to be large, because there will usually be one or more groups of 1 or even zero length, even a reference of 1 or 2 could significantly reduce the bits needed. (This method of representing groups is better than the secondary bit map feature of Complex in the original GRIB, except in the somewhat pathological case of one or a few very large groups and many very small groups.) Octet 47 is zero for row-by-row packing. [MDL software does not support row by row packing.]

[23]Octets 48 and 49 apply to Template 5.3 (specified in octets 10-11).

[24][First-order differencing is not supported by PK_GRIB2 and UNPK_GRIB2. When

Octets	Definition		
	Simple	Complex	Spatial Differencing
49	-----	-----	Field width in octets of each of the extra descriptors needed for spatial differencing (see octets immediately following octet 5 in Section 7

3. GRIB2 Section 6

Contents of Section 6. When "same" appears in a box, it means the same as the box to the left. All octets will be filled by PK_GRIB2; the user need not furnish values in IS6( ).

Octets	Definition		
	Simple	Complex	Spatial Differencing
1-4	Length of section in bytes[1]	Same	Same
5	Number of section = 6	Same	Same
** 6	Bit map indicator[2]: 0 = A bit map is included in this section 1-253 = A bit map of this number applies, but is not included in the message[3] 254 = A bit map applies to this grid, but is not included; the most recently encountered previous bit map in	Same[5]	Same

a user asks for Spatial Differencing, MDL code determines whether or not second-order spatial differencing is space-efficient; if it is, second-order differencing is used; if not, Complex is used. The method of determining this is rather crude, but is relatively fast and good enough for the purpose.]

[25][PK_GRIB2 does not support second-order differencing when there is a secondary missing value. It is assumed a secondary missing value will be rare, and when it is present it would be for a field that is not conducive to second-order differencing.]

[1]Inclusion of Section 6 is mandatory. When a bit map is not present, this value will be 6.

Octets	Definition		
	Simple	Complex	Spatial Differencing
	this message applies.[4] 255 = No bit map		
7-	Bit map[6]	Same[7]	Same

[2]Note that when a bitmap is being dealt with, only one kind of missing value is possible, which would be called the primary missing value.

[3]Numbers 1-253 are used for previously defined and numbered bit maps. [PK_GRIB2 will accept a grid with the missing values already removed together with a bit map number; UNPK_GRIB2 will return that same grid and the bit map number.] (This partially invalidates the concept that each grid in the message be fully defined.)

[4]This capability allows a bit map to be in the message only once, but apply to all grids in the message. This might be advantageous when a model does not provide output for all points in a rectangular grid.

[5][PK_GRIB2 does not support a bit map with Complex or Spatial Differencing. Therefore, the only number here will be 255 unless Simple packing is being used. If a bit map is provided and the user specifies Complex or Spatial Differencing, missing values (octets 24-27 in Section 5) are inserted into the grid before packing. UNPK_GRIB2 recognizes a bit map supplied with a data field that was packed with either the Simple or the Complex methods. When a bit map is present in a GRIB2 message, UNPK_GRIB2 gives the user the option of receiving the the unpacked data field with or without missing values embedded in it. With the Simple method, if the user wants a data field with missing values in it, then the missing value to use must be supplemented through the MISSP (integer) or XMISSP (float) arguments of UNPK_GRIB2. With the Complex method, the primary missing values are stored within the GRIB2 message in octets 24-27 of Section 5. If no bit map is present within a GRIB2 message, then the user will either receive an unpacked data field without missing values in it (Simple method) or a data field with primary and/or secondary missing values embedded within it (Complex and Spatial Differencing methods). With the Complex and Spatial Differencing methods, octet 23 of Section 5 can be tested to determine if there are primary and/or secondary missing values in the data field.]

[6]This will be a string of NX*NY zeros and ones, each representing a gridpoint in the same order as the grid of values provided. A “1" means the gridpoint has a non-missing value; a “0" means the value for the gridpoint is missing.

[7][A bit map is not used by PK_GRIB2 in Complex or Spatial Differencing. If a bit map has been provided to PK_GRIB2 along with a grid of values with the missing values removed, it must be accompanied by value of 1 or 2 in octet 23 of Section 5 (the missing value management flag) and a user-supplied value in octets

24-27 of Section 5 (the primary missing value substitute). See also footnote 5.]

4. GRIB2 Section 7

Contents of Section 7. When "same" appears in a box, it means the same as the box to the left. All octets will be filled by PK_GRIB2; the user need not furnish values in IS7( ).

Octets	Definition		
	Simple	Complex	Spatial Differencing
1-4	Length of section in bytes (nn)	Same	Same
5	Number of section = 7	Same	Same
6-nn	Data values (X)[1]	-----	-----
6-ww	-----	-----	The first N non-missing values in the field (after scaling), where N is the order of differencing followed by the minimum of the differences[2]
-xx[3]	-----	NG reference values for the data values (X1)[4]	Same

[1]Each data value (X) is scaled according to the formula

$$X = (Y * 10^{D} - R) * 2^{-E}$$

where Y is the original value [see Section 5 for definitions of R (octets 12-15), D (octets 18-19), and E (octets 16-17)], and occupies the number of bits specified in Section 5, octet 20. To emphasize, R is the minimum value in the field after multiplying the values by 10^{D}. When there are no missing values, there are NX*NY values. When there is a bit map, there will be the number of values specified by Section 5, octets 6-9. The original data values can be recovered by the formula

$$Y = [R + (X * 2^{E})] * 10^{-D}$$

[2]For instance, for 2nd order differencing, there would be three values. Even though the overall minimum has already been removed (octets 12-15 in Section 5), the differences can be negative or positive. To make them all positive, the minimum is subtracted from each value and that minimum placed here. This minimum will undoubtedly be negative and will be represented by the leftmost bit being 1 as is usual for negative numbers.

[3]Follows octet 5 for complex or octet ww for spatial differencing.

Octets	Definition		
	Simple	Complex	Spatial Differencing
(xx+1) -yy	-----	NG widths for the group widths[5]	Same
(yy+1) -zz	-----	(NG-1) lengths of the groups[6]	Same
(zz+1) -nn	-----	Data values (X2)[7]	Same[8]

[4]Each of these reference values (X1 in the scaling formula) occupy the number of bits specified in Section 5, octet 20. NG, the number of groups, is specified in Section 5, octets 32-35.

[5]Each of these values occupies the number of bits specified in Section 5, octet 37, and is the number of bits necessary to specify the values in the corresponding group (width of group) AFTER the reference value specified in Section 5, octet 36, has been removed (subtracted).

[6]Each value occupies the number of bits specified in Section 5, octet 47, and is the number of values in each group AFTER the reference value specified in Section 5, octets 38-41, is removed (subtracted). Note that this is (NG-1) because the true length of the last group is stored in octets 43-46 of Section 5.

[7]Each data value (X2) is scaled according to the formula

$$X2 = (Y * 10^{D} - R) * 2^{-E} - X1$$

where Y is the original value [see Section 5 for definitions of R (octets 12-15), (octets 18-19), and E (octets 16-17)] X2 occupies the number of bits specified in Section 5, octet 37. The NG values of X1 are the group reference values specified in this section, ending with octet xx). Primary missing values, when there are any, are carried as the largest possible value in the field (all ones), and secondary missing values, when there any, are carried as the largest possible value in the field minus 1 (the least significant bit being zero and all bits preceding that being ones). There is always NX*NY values. The original values can be recovered by the formula

$$Y = [(R + (X1 + X2) * 2^{E}] * 10^{-D}$$

[8]Even when there are only NX*NY-1 first-order differences [not supported by MDL software] and only NX*NY-2 second-order differences, there are always NX*NY values present. The first N, where N is the order of differencing, non-missing values are "dummy" and are set to zero. This allows reasonably easy accommodation of missing values. For instance for second-order difference packing, if the second value in the original grid is missing, and the first and third values are legitimate, when packed the first and third values will be

"dummy," the second will be represented as missing, and the fourth will be the first second-order difference.