TMAP'S "GT" DATA FILE FORMAT

written 09/88 by Mark Verschell

The "GT" data file format ('Grids-at-timesteps') is 1, 2, or 3 dimensional spatial grids of data of 1 or more variables at specific timesteps. Each variable can be defined on the same spatial grid, or defined on different spatial grids. However each variable must be defined at each timestep. In addition, the data may be contained in multiple "disk files" with each disk file starting and ending with a complete timestep.

The GT data file's spatial and temporal structure is defined in a descriptor file (see the associated document - ANNOTATED.DES - for documentation of the descriptor file). The descriptor file provides all of the information necessary to use a GT data file.

A GT data file is an unformatted direct-access (i.e fixed record length) file. The minimum record size is 32 bytes, the maximum is that allowed by VMS record-length restrictions. The GT data file contains 2 kinds of records : "STEP" records and "variable" records. NOTE: The "STEP" record and the 'variable' record must be the same length, therefor the minimum 'variable' record is 6 values. See the accompanying graphic at the end of this chapter.

A "STEP" record precedes each group of variables in a timestep. Its structure is as follows:

```
         1         2         3         4         5         6         7
12345678901234567890123456789012345678901234567890123456789012345678901234567890123456 ...
STEPaaaaaaaabbbbccddeeeeffffgggg ... (g's to end of record)
```

where:

- STEP = The string 'STEP'
- aaaaaaaa = REAL*8 value containing the timestep
- bbbb = INTEGER*4 value containing the total number of records in this timestep - including the "STEP" record
- cc = INTEGER*2 value containing the number of variables in this timestep
- dd = INTEGER*2 value - unused
- eeee = CHARACTER*4 value containing the data identification code
- ffff = CHARACTER*4 value containing the modification code of this data identification code
- gggg = INTEGER*4 fill to end of record (record length needs to be evenly divisible by 4)

Remember that this is an unformatted file, i.e. the "STEP" record would be output with a WRITE statement similar to the following:

```
        WRITE (10) 'STEP',tstep,totrecs,totvars,dummy,runnum,modnum,
     .    (fill(i),i=1,leftover)
```

The gggg field can alternatively be an array of any 4, 8, 12 ... byte type. Then it could be used for storing of optional information. For example, a character string that stores information about the data. The total record length in bytes still must be evenly divisible by 4.

The 'variable' records store the actual data. The structure is as follows:

```
         1         2         3         4         5         6         7
12345678901234567890123456789012345678901234567890123456789012345678901234567890123456 ...
vvvvssss a1  a2  a3 ...
```

where:

vvvv = CHARACTER*4 string containing the code name for a variable, e.g. 'TEMP'
ssss = INTEGER*4 value containing the sequence number for a variable (explained below)
a1
a2 = REAL*4 values containing the data
a3

The data are stored in the GT file in the order specified in the descriptor file. Remembering that the data are stored in a 1, 2, or 3 dimensional spatial grid, each record contains the data for 1 complete "vector" of data. More explicitly, if the grid is one dimensional, all the data are stored in 1 record. If it is two dimensional (m * n), then the data are stored as n records of m values or m records of n values. Lastly, if it is three dimensional (m*n*p), then the data are stored as p groups of m records of n values (xyz ordering), or p groups of n records of m values (yxz ordering), etc... (all possible permutations of xyz).

For each variable, the range of data stored in the file may be any 1, 2, or 3 dimensional, rectangular sub-region of the full 1, 2, or 3 dimensional defining grid for that variable. For example, if a variable were defined on a grid whose size was 200 x 100 x 50 in X, Y, and Z respectively a GT data set might contain reduced limits in (say) X and Z such that only 51:75 x 100 x 1:1 (25 x 100 x 1 points) were stored.

The sequence number is the position of the record within the full defining grid for that variable. For example, in a 3-D grid of xyz-ordered data (each record is a "vector" of data along the x axis) a record which is number j (of a possible m) counting from the southern boundary and number k (of a possible p) counting from the surface would have the sequence number of (k-1)*m + j. If the ordering in the same example had been xzy the sequence number would be (j-1)*p+k.

There is a picture of the file setup located with this document.