TMAP'S GRID FILE LAYOUT

The grid file is used to describe axes and grids to programs that use TMAP library of routines (e.g. FERRET). It can describe all manner of axes from regular to irregular; modulo and non-modulo; spatial and time; and grids that use these axes. This document will describe how to build a grid file that will correctly describe the grid arrangement you desire.

The grid file has four (4) parts to it. Not all are required in any one grid file. We will go through each of the parts and describe them with examples, and talk about some of the inter-relationships.

First, let's cover some important things about the grid files in general. The files are "FREE" format. This means on any one line (except for COORDS definition - see below) any of the items in a definition can be located anywhere on that line, as long as they are in the correct order, do not extend past one line, and are not longer than 132 characters. This allows you to "pretty up" the file anyway you like. LINE definitions have to come before any GRID, TIME, or COORDS definitions that reference them. GRID, TIME, and COORDS definitions can come in any order. The usual order for a grid file is LINE, GRID, TIME, COORDS. It is a good idea to leave the explanatory documentation in a grid file in there when you copy it (the section enclosed in *'s). These will be useful for quick reference should you ever forget what components belong in each definition. Lastly, before starting with a part by part break-down, all lines that begin with a * are considered comments (do not embed comments in COORDS definitions!).

I. LINE - defining axes

This part of the grid file is used to define an axis or axes. Here is an example:

```
*****************************************************************************
* Line definitions first: name, number of points, ordering, and defined by
* start,delta or coordinate values (given below) "ordering" may be "WE"
* (longitude), "SN"(latitude), "UD"(depth), "TI"(time) or "NA"(non-standard
* values)
*****************************************************************************
LINE   XAXIS    10     WE    LONGITUDE    START,DELTA:      250.    1.
LINE   YAXIS     5     SN     LATITUDE    GIVEN_BELOW:
LINE   ZAXIS    11     UD       METERS    GIVEN_BELOW:
LINE   TAXIS   120     TI        HOURS    START,DELTA:    17592.   73.   MODULO
```

First comes "LINE" that identifies this line as an axis definition, then comes the unique line name (e.g. XAXIS). The number of points in the axis is next, followed by the line orientation (e.g. WE="west to east"; SN="south to north"; UD="up to down, or depth - increasing downwards"; TI="forward time"). After this comes the units of the axis, LATITUDE, METERS, HOURS... The next component tells how the axis is defined. There are two (2) choices: regular axis are defined by "START,DELTA:" followed by the starting point in user units, and the distance between two points on the axis in user units; and "GIVEN_BELOW:" which indicates that the axis points are irregularly spaced and will be explicitly defined in a COORDS definition (defined below). The starting point and delta should be real numbers. The last item is optional. It is the MODULO flag, which indicates whether an axis is modulo (i.e. one which wraps around on itself, as in LONGITUDE defined around the globe). If the line is not modulo, this parameter is omitted. If an irregular axis (GIVEN_BELOW:) is MODULO, 2 "dummy" parameters need to be put between the "GIVEN_BELOW:" and the "MODULO". For example: GIVEN_BELOW: x x MODULO. The x's will be ignored.

II. GRID - defining grids from axes.

This part of the grid file is where grids are defined from >pre-existing This is an important point. If an axis name is not defined within a grid file and it is used in a grid definition it must have been already defined. There are 2 ways an

axis could be already defined: i) it could be pre-defined by the application you are running (e.g. "FERRET"); or ii) by a separate grid file that has already been processed. Otherwise the program will generate an error. Here is an example of grid definitions:

```
*********************************************************************************
* Next comes grid definitions: gridname and rotation, followed by lines they
* contain (along with inner/outer product flag)
*********************************************************************************
GRID  G1  0.0   XAXIS   OUTER   YAXIS  OUTER   NORMAL   OUTER   UNKNOWN   OUTER
GRID  G2  0.0   XAXIS   OUTER   YAXIS  OUTER   ZAXIS    OUTER   UNKNOWN   OUTER
GRID  G3  0.0   XAXIS   OUTER   YAXIS  OUTER   ZAXIS    OUTER     TAXIS   OUTER
```

First comes "GRID" to indicate that this line is a grid definition, then the unique name for the grid you are going to define. Next is the grid rotation in the xy plane, a real number indicating the number of degrees the grid is rotated from a normal world-coordinate system (e.g. Latitude, Longitude on earth). Rotation is not currently supported, and should be 0.0. Next comes the four (4) axes used to define the grid paired with the type of "product" they are in relation to the other axes. These are in order x-axis, y-axis, z-axis, t-axis. Product can be either INNER or OUTER. OUTER products define a regular grid, where points can be located at any combination of valid points along the defined axis, independent of location on the other axes. INNER products are grids defined by ordered pairs/triplets/quadruplets. In effect they are discrete points located in time-space. Therefor, OUTER products define a regular, rectangular region in space-time, a cube for example, while INNER products define a scatter of points in a "fuzzy" but definable region (INNER are products are not currently supported as of 6/90). In our example above, three (3) grids are defined: a x-y grid with a NORMAL Z-axis, and time left to be determined by the data set descriptor (see documentation on descriptors); a x-y-z grid also with an UNKNOWN time axis; and a fully defined x-y-z-t grid.

III. TIME - giving a time axis an initial date

The TIME axis is the simplest of the four parts. It's only function is to locate the starting point of a time axis in "World" coordinates. Time axis are identified in two ways: time step values - which are defined in the LINE section, in our above example that is starting with 17592 with 73 unit increments; and "calender" time, for example 12-DEC-1989-10:35:15. Here is an example:

```
*********************************************************************************
* Next time line definitions: timeaxis linename, followed by t0
*********************************************************************************
TIME    TAXIS      14-JAN-1980:14:00:00
```

As you can see the only things in this definition are "TIME", the name of the time axis being defined, and the date for timestep 0. Notice this is not necessarily the first time step you defined in the LINE section. In our example the first time step is 17592, however the date given here is for time step 0. This is an important point. The t0 date must be in the above format. That is 2 digits for day (0 filled - e.g. 04), "-", a 3 character month, "-", a four digit year, ":", 2 digits for hour (0 filled), ":", 2 digits for minute (0 filled), ":", and 2 digits for second (again, 0 filled).

IV. COORDS - defining locations on an irregular axis

This is where the coordinates for irregular lines are explicitly defined. This is the most complicated part of a grid file, and requires extra care in preparation. It has two parts: the first is the COORDS definition line which describes the format of the irregular axis and the starting point of the first

box; the second part is pairs of coordinates and box sizes. Some explanation of how an axis is set up is required here.

When you define a regular axis ("START,DELTA:" in the LINE section) you are actually describing grid point locations and a box containing each grid point. Because the line is regular these boxes are all the same size and are symmetrical around the grid point. A picture will illustrate this:

```
|   x   |   x   |   x   |   x   |   x   |   x   |   x   |
```

Here the x's designate grid points you have defined on a regular axis and the |'s are the implicitly defined borders of boxes containing the grid points. This works great for regular axes, but in irregular axes there is not enough information given by just the locations of the grid points to determine where the box boundaries go. Again, a picture will help clarify this:

```
|     x   |   x   |  x  |  x  | x | x |x|x| x | x |  x  |  x  |   x   |  x    |
```

As you can see the boxes are all different size and the end boxes are not symmetrical around the grid points. This is a very simple irregular axis and for more complex cases it might be the rule rather than the exception that the grid boxes are not symmetrical around the grid points. Therefor, an irregular axis is defined with three (3) pieces of information. The locations of the grid points, the location of the starting point of the first grid box, and the size of each box. here is an example of a COORDS section:

```
*****************************************************************************
* Lastly, coordinate/box size pairs for those lines not specified by
* start,delta: line name followed by format for coordinates followed by start
* location of first grid box
*****************************************************************************
COORDS     YAXIS      (5(F8.0,F6.0))      -9.50
   -7.90    3.   -5.00    3.   -2.00   3.5    2.00   3.5    5.00    3.
COORDS     ZAXIS      (7(F5.0,F4.0))      0.00
  0.00 10.  20.00 20.  40.00 20.  60.00 20.  80.00 20. 100.00 20. 120.00  20.
140.00 30. 180.00 80. 300.00 90. 500.00 95.
```

Only the YAXIS definition will be explained, since the ZAXIS definition is no different in concepts, just in detail. First comes the COORDS line which contains "COORDS", the name of the irregular axis to be defined, the format for the data (explained below), and the location of the first grid box (in the same units as the locations of the grid points). The lines below that (up to the next COORDS definition) are the lines containing the grid point locations matched one-to-one with the size of the grid box the grid point is located in.

The format can be constructed in a number of ways. The only rules concerning it are that it must be enclosed in parenthesis, it should describe the data on one line only, and every line, except the last need to contain the same number of data pairs. The last line can contain either the same or less number of data pairs. The recommended way is to follow the form (n(Fq.0,Fr.0)) where n is the number of pairs per line, q is the number of spaces the grid point location is defined in, and r is the number of spaces the box size is defined in. In the YAXIS definition this is 5 pairs per line, there are 8 spaces reserved for the grid point location, and there are 6 points reserved for the box size. The full layout of the YAXIS line is:

Grid Point Location	Box size	Box Start Location
-7.90	3.0	-9.50
-5.00	3.0	-6.50
-2.00	3.5	-3.50
2.00	3.5	0.00
5.00	3.0	3.50

With the last box ending at 6.50.