NTv2

National Transformation

Version 2

Developer's Guide

D.R. Junkins
S.A. Farley

Geodetic Survey Division
Geomatics Canada

September 1995

Copyright

Table of Contents

1. Introduction

This document provides complete specifications for the hybrid grid shift file structure used in Version 2 of the National Transformation, and the procedures used to access and apply the data it contains. The two major differences in the data provided by Version 2 are the inclusion of accuracy estimates for each of the predicted shift values, and grids of greater density in selected areas, corresponding to high densities of control survey networks.

The file structure devised for Version 2 allows multiple grid shift files at various densities. Each of these base files can have areas of densified grids, which in turn can themselves have densified areas. All of these grid areas are combined as sub-files of a single larger file. This hybrid file structure is transparent to the user because the selection of the appropriate grid area is automatic. Similar to Version 1, a utility to extract a smaller file for a selected area addresses the concerns that some users may have about file storage space or geographical area of operation.

This documentation is intended for third-party developers who are licensed to incorporate the NTv2 as a feature of their products, or for end users who integrate the NTv2 into their own systems or perform other custom modifications for in-house use. For direct users of the NT in its unaltered form, this document may be treated as non-essential background material, since it is not critical to understand it in order to use the system effectively. Version 1 specifications are also included for comparison.

A descriptive overview of the file structures and how they are used is given in Sections 2, 3, 4, and 5. Section 2 describes the elementary grid shift file as used in both Versions 1 and 2, with minor differences. Section 3 describes the hybrid grid shift file used in Version 2, with particular attention to the methodology of grid densification. Section 4 gives a description of the three essential processes to access the grid shift files, and Section 5 suggests various strategies for implementation into the developer's environment. Detailed descriptions of the three processes are given next. Finally, Appendices A and B contain detailed specifications of the record structures for Versions 1 and 2, respectively.

2. Elementary Grid Shift File Description

The National Transformation Version 1 was based on a grid shift file that is strictly regular in structure. It must cover a rectangular area bounded by unique values for each of the following: lower latitude, upper latitude, lower longitude, and upper longitude. The grid intervals are specified separately for latitude and longitude, and are fixed at these values throughout the file. Indexing of grid intersection points begins at the lower right-hand, or south-east, corner of the grid rectangle (Figure 1).

```
    Figure 1:  Grid Limits and Point Indexing
    -----------------------------------------


              Longitude
              Interval
                dLon
              #-------#
                                             Upper
      +-------+-------+-------+-------+  -- Latitude
   m*n|       |       |       |       |      Limit
      |       |       |       |       |        N
      |       |       |       |       |
      +-------+-------+-------+-------+
      |       |       |       |       |
      |       |       |       |       |
      |       |       |       |       |
      +-------+-------+-------+-------+  #
    3n|    ...|   2n+3|   2n+2|   2n+1|  |  Latitude
      |       |       |       |       |  |  Interval
      |       |       |       |       |  |    dLat
      +-------+-------+-------+-------+  #
    2n|    ...|    n+3|    n+2|    n+1|
      |       |       |       |       |
      |       |       |       |       |      Lower
      +-------+-------+-------+-------+  -- Latitude
     n    ...       3       2       1        Limit
                                               S

      |                               |
    Upper                           Lower
  Longitude                       Longitude
    Limit                           Limit
      W                               E

n = number of columns          m = number of rows


             W - E                          N - S
n = integer( ----- ) + 1       m = integer( ----- ) + 1
              dLon                           dLat
```

This type of structure allows the direct computation of the record indices for the four corners of the cell containing a point to be processed (Figure 2). Since no search algorithms are required, speed of retrieval is independent of file size. It also permits the reduction of file size by creating a subset covering a smaller rectangular area within the original, suiting users with a more regional interest.

```
          Figure 2:  Grid Cell Direct Indexing
          ------------------------------------


                          jth
                        column
                           |

           +-------+-------+-------+-------+
           |       |       |       |       |
           |       |       |       |       |
           |       |       |       |       |
           +-------•-------•-------+-------+
           |      D|      C|       |       |
           |       |   •   |       |       |
           |       |  P    |       |       |
           +-------•-------•-------+-------+  -- ith row
           |      B|      A|       |       |
           |       |       |       |       |
           |       |       |       |       |
           +-------+-------+-------+-------+
           |       |       |       |       |
           |       |       |       |       |
           |       |       |       |       |
           +-------+-------+-------+-------+


             Lat(P) - S
i = integer( ---------- ) + 1         A = n(i-1) + j
                dLat
                                      B = A + 1

             Lon(P) - E               C = A + n
j = integer( ---------- ) + 1
                dLon                  D = C + 1
```

The size of the file, as determined by the area it covers and the density of the grid, does not affect the software used to access and apply the data. All of the information required to describe the data is contained in a block of header records at the beginning of the file.

The header records are followed by the grid shift records which contain the predicted shifts at each point in the grid. The last record is a flag to indicate the end of file. Appendix A provides a more detailed description of the grid shift file as used in Version 1.

This elementary grid shift file is also used as the fundamental building block for Version 2. The major difference is that while the Version 1 grid shift records have two double precision variables for the latitude and longitude shifts, the Version 2 grid shift records contain four single precision values for the two shifts and their estimated accuracies.

3. Hybrid Grid Shift File Description

The hybrid grid shift file used for Version 2 of the National Transformation combines one or more base grids of various densities with grids of greater densities covering designated areas within each of the base grid areas. Different areas within the same base grid may be densified using different grid intervals, and each of these areas can themselves have areas that are further densified. Each grid area is compiled separately, and then all of the elementary grid shift files are combined into a single hybrid grid shift file. The rules for defining and selecting grid areas ensure that a unique determination will be made for any point to be transformed, with no dependency on the user to select the appropriate grid.

Rules for Densification

As with the specifications of the elementary grid shift file, a set of rules must be followed to enable the direct computation of the record indices for the four corners of the cell containing a point to be processed. These rules ensure that points falling on or near the densification boundary will have unique values, with no discontinuity between the parent and densified grid areas.

The first rule concerns the extent and grid intervals of the densified area (Figure 3):

- 1- i) The densified area must be rectangular, with the extent in each dimension being a whole number (integer) of parent grid intervals.
- 1- ii) Densified grid intervals must be a whole number (integer) of divisions of the parent grid intervals.
- 1- iii) Densified area boundaries must be coincident with the parent grid, enclosing only complete cells.
- 1- iv) Densified areas may not overlap each other.

```
Figure 3:  Valid Densified Grid Areas
-------------------------------------


        Area A (valid)
 +-------+ + + + +-------+-------+
 |       + + + + +       |       |
 |       + + + + +       |   +++++  Area C
 |       + + + + +       |   +++++ (not valid)
 +-------+ + + + +-------+---+++++
 |       + + + + +       |       |
 |       + + + + +       |       |
 |       + + + + +       |       |
 +-------+ + + + +-------+-------+
 |       |       |       |       |
 |       |       |   + + + + + + +
 |       |       |   + + + + + + +
 +++++++++-------+---+ + + + + + +  Area D
 +++++++++       |   + + + + + + + (not valid)
 +++++++++       |   + + + + + + +
 +++++++++       |   + + + + + + +
 +++++++++-------+---+ + + + + + +
```

```
  Area B +++++++++
 (valid) +++++++++
```

```
Areas A and B are valid

- boundaries are coincident with parent grid cells


Areas C and D are NOT valid

- not all boundaries are coincident with parent grid cells
```

The second rule concerns the shift values assigned to the densified grid points (Figure 4):

2- i) All grid points falling within the interior of the densified area have shifts determined by the model being implemented, i.e. the same one used to compute the parent grid.

2- ii) For grid points on the perimeter, the shift values are determined by linear interpolation between adjacent parent grid points.

2- iii) For grid points on a common perimeter segment between two adjoining densified areas of the same density, the shifts determined by the model (as in 2-i) above) are retained.

2- iv) For grid points on a common perimeter segment between two adjoining densified areas of different density, the shifts determined by the model (as in 2-i) above) are retained for the less dense grid, while the shift values for the more dense grid are determined by linear interpolation of the less dense grid. (This rule also applies to base grids of different densities.)

```
Figure 4:  Values at Perimeter Points
-------------------------------------



 P       P       P       P       P



 P       P # # # P # # # P       P
         # + + + + + + + #
         # + + + + + + + #
         # + + + + + + + #
 P       P + + + P + + + P       P
         # + + + + + + + #
         # + + + + + + + #
         # + + + + + + + #
 P       P # # # P # # # P       P



 P       P       P       P       P



P  - parent grid points
   - retain original values

+  - densified grid points
   - computed from same model as parent grid

#  - densified grid perimeter points
   - interpolated from parent grid
```

File Construction

The hybrid file is a series of sub-files, each of which is a complete elementary file with header records and grid shift records. The header records contain information that is specific to the individual sub-file, such as file identifiers, creation and revision dates, grid limits and intervals, and the number of grid shift records.

The order of the sub-files is of no consequence, since an index table of the sub-files is built each time the hybrid file is opened. As with the elementary grid shift file, the order of the records within each sub-file is critical.

At the beginning of the hybrid file is an overview header block of information that is common to all sub-files. This includes the number of header records in the overview and sub-file header blocks, the number of sub-files, grid shift units, a version identifier, and reference system names and ellipsoid parameters. Overview header data is cross-checked during file construction.

At the end of the hybrid file is an end of file record signifying that there are no more sub-files.

4. Accessing the Hybrid Grid Shift File

To use the data from the grid shift file in a program, there are three steps involved. The first is to build an index of its contents each time the file is opened. Subroutine TABLE does this by reading the pertinent information from the overview header block and all of the sub-file header blocks. This provides a map in memory that can be quickly searched each time a coordinate is submitted for processing.

The second step is the search function performed by subroutine FGRID. Its algorithm is designed to uniquely select the appropriate sub-file, with particular care taken to ensure that the most dense grid containing the point is used. It also identifies cases in which the point falls on the upper limits of the area covered by each sub-file, so that special steps may be taken to handle it.

The third step is to select from the specified sub-file the records which correspond to the four corners of the cell containing the submitted point, and perform the bilinear interpolation to compute the shifts and their accuracies that apply to that point. This is done by subroutine GRIDINT.

Each of these subroutines is presented in the following sections using a mix of descriptive narrative, pseudo-code, and data dictionary style terminology. Many of the terms refer to the elements of the hybrid grid shift file detailed in Appendix B, while others are local to the processes themselves. Appendix A describes the file structure for Version 1, and is provided only for comparison by developers who have implemented this earlier format.

5. Implementing the Hybrid Grid Shift File

There are many ways in which a developer might adapt the NTv2 grid shift file and procedures for custom applications.

Some of the simpler approaches would be:

1) Port the software from DOS to another type of computer system, such as UNIX, VAX/VMS, or Mac/OS. The User's Guide section on program READDA gives details on this procedure.

2) Change the formats for the input and output records to conform to the user's requirements. The appropriate statements are found in subroutines GETGEO, GETUTM, GEOOUT, and UTMOUT.

3) Devise an operating-system-level procedure to convert the application to a total batch process for repetitive or "blind" application. The procedure would launch program INTGRID and supply all the necessary keystrokes to emulate an interactive session which selects and responds to the required options.

A more integrated approach would be to use the three subroutines TABLE, FGRID, and GRIDINT, to create an NTv2 feature of another existing system, such as a GIS or mapping or coordinate conversion package. These three routines form the heart of program INTGRID - the rest is user interface and input/output function. By using these routines, the end-users of the existing system would be provided with a standard conversion tool built in to the system with which they are already familiar. It would act directly on the data structures of that system, and not require any export/import activity, with its potential for problems.

The use of these routines is demonstrated in a special developer's program called NOFRILLS. This program operates in batch mode, with only the minimum of code required to open the appropriate files, call the three routines, and close the files on completion.

Subroutine TABLE

Build Grid_File_Index Tables

Subroutine TABLE builds the arrays of data that represent the Grid_File_Index by reading all of the Overview_Header_Block records and Sub_File_Header_Block records each time the Grid_Shift_File is opened.

To access the information from the various header blocks, the following algorithm is used:

1) The Overview_Header_Block at the beginning of the file provides the:

   Number_of_Overview_Header_Records
   and
   Number_of_Sub_File_Header_Records in each Sub_File
   and
   Number_of_Sub_Files

2) The Sub_File_Header_Block for each Sub_File provides the:

   Grid_Shift_Record_Count

3) The number of records preceding the start of the first Sub_File is:

   Sub_File_1_Offset= Number_of_Overview_Header_Records

4) The number of records preceding the start of the second Sub_File can be computed as:

   Sub_File_2_Offset= Number_of_Overview_Header_Records
   + Number_of_Sub_File_Header_Records
   + Sub_File_1_Grid_Shift_Record_Count

5) and subsequently, for each i^{th} Sub_File:

Sub_File_i_Offset	=	Sub_File_(i-1)_Offset
	+	Number_of_Sub_File_Header_Records
	+	Sub_File_(i-1)_Grid_Shift_Record_Count

The header block information is stored in arrays which are returned to the calling routine as formal parameters. The following lists indicate the data elements and the array variable names in which they are placed. More detail on variable types and length can be found in the file description. An array is also created for the Sub_File_Offsets as determined by the preceding algorithm.

From the Overview_Header_Block:

	Data Element Name	*Variable Name*
i)	Number_of_Overview_Header_Records	NORECS
ii)	Number_of_Sub_File_Header_Records	NSRECS
iii)	Number_of_Sub_Files	NFILES
iv)	Grid_Shift_Data_Type	TYPOUT
v)	Version_Number	VERSION
vi)	System_From	FDATUM
vii)	System_To	TDATUM
viii)	Semi_Major_Axis_From	FELLIPS(1)
ix)	Semi_Minor_Axis_From	FELLIPS(2)
x)	Semi_Major_Axis_To	TELLIPS(1)
xi)	Semi_Minor_Axis_To	TELLIPS(2)

From the Sub_File_Header_Block for each of the sub-files:

	Data Element Name	*Variable Name*
i)	Sub_File_Name	ANAMEG(I)
ii)	Parent_File_Name	APGRID(I)
iii)	Creation_Date	(not used)
iv)	Last_Revision_Date	(not used)
v)	Lower_Latitude_Limit	ALIMIT(I,1)
vi)	Upper_Latitude_Limit	ALIMIT(I,2)
vii)	Lower_Longitude_Limit	ALIMIT(I,3)
viii)	Upper_Longitude_Limit	ALIMIT(I,4)
ix)	Latitude_Grid_Interval	ALIMIT(I,5)
x)	Longitude_Grid_Interval	ALIMIT(I,6)
xi)	Grid_Shift_Record_Count	AGSCOUNT(I)

As computed from the algorithm described above:

Data Element Name	*Variable Name*
Sub_File_i_Offset	ASTART(I)

Subroutine FGRID

Find Grid_Shift_Sub_File

The proper Grid_Shift_Sub_File for any point to be processed is defined as the one:

- whose limits enclose the point,
    and
- which has no Densified_Grid_Shift_Sub_File whose limits also enclose the point.

The rules for defining grid areas prohibit a point from falling within the limits of more than one Grid_Shift_Sub_File at the same level, making the selection unique. The basic rule is that Grid_Shift_Sub_Files with the same Parent_Grid_Name cannot have overlapping boundaries.

If a point is **on** the limits of a Grid_Shift_Sub_File, then special conditions apply:

- if it is on the **lower** limits of latitude or longitude, it is considered as **within** the limits
- if it is on the **upper** limits of latitude or longitude, it is considered as **outside** the limits

This definition prevents a point from belonging to two or more Grid_Shift_Sub_Files that have adjoining boundaries and the same Parent_Grid_Name. The only exception is a point which is on either of the upper limits of a Grid_Shift_Sub_File, but not within any other. It will be considered as within that Grid_Shift_Sub_File.

The steps in the process are:

1. The Current_Grid_Name is initially set to 'NONE', which is the Parent_Grid_Name for all of the Base_Grid_Sub_Files.

2. All Grid_Shift_Sub_Files having a Parent_Grid_Name the same as the Current_Grid_Name are tested to determine whether any of them enclose the point.

3. If any Grid_Shift_Sub_File does enclose the point, its Sub_File_Name becomes the Current_Grid_Name, and Step 2 is repeated.

4. If no further Grid_Shift_Sub_File encloses the point, the Sub_File corresponding to the Current_Grid_Name is selected, and its Sub_File_Number is returned.

5. If no Grid_Shift_Sub_File encloses the point, a Sub_File_Number of zero is returned.

6. A flag indicating whether the point is on either of the upper limits is also returned. The flag values are set as follows:
   [1] point is within the limits
   [2] point is on upper latitude limit
   [3] point is on upper longitude limit
   [4] point is on both upper latitude and upper longitude limits

   A point may fall into one of these categories for one grid area, but may also fall into another of the categories for an adjacent grid area if both Grid_Shift_Sub_Files have the same Parent_Grid_Name. A point may even fall into three or four categories if it is on the junction of three or four adjoining grid areas. In these cases of multiple selections, the Grid_Shift_Sub_File corresponding to the lowest value of the flag is selected by default.

Subroutine GRIDINT

Grid_Shift_Interpolation

1. Retrieve Grid Shifts

If a specific grid point is to be retrieved, then the Record_Index of that point is computed by the following sequence (see Figures 1 & 2 in Section 2: Elementary Grid Shift File Description):

$$\text{Row\_Index} = \frac{\text{Point\_Latitude} - \text{Lower\_Latitude}}{\text{Latitude\_Grid\_Interval}} + 1$$

$$\text{Column\_Index} = \frac{\text{Point\_Longitude} - \text{Lower\_Longitude}}{\text{Longitude\_Grid\_Interval}} + 1$$

$$\text{Grid\_Points\_Per\_Row} = \frac{\text{Upper\_Longitude} - \text{Lower\_Longitude}}{\text{Longitude\_Grid\_Interval}} + 1$$

$$\begin{aligned}\text{Record\_Index} = & \ (\text{Row\_Index} - 1) * \text{Grid\_Points\_Per\_Row} \\ + & \ \text{Column\_Index} \\ + & \ \text{Sub\_File\_Offset}\end{aligned}$$

All division operations are truncated to integers. Thus, if the point coordinates are not exactly identical to coordinates of a grid point, the Record_Index will correspond to the lower right-hand corner of the grid cell which contains the point.

This means that if a point is on the lower limit of the cell, it is considered as in the cell, but if it is on the upper limit, it is considered as being in the next adjoining cell.

If the four grid points of the corners of the cell are to be retrieved, then the same sequence as above is used to first determine the Record_Index_LR. The Record_Indexes for each of the other three corners are determined as:

Record_Index_LL = Record_Index_LR + 1

Record_Index_UR = Record_Index_LR + Grid_Points_Per_Row

Record_Index_UL = Record_Index_UR + 1

If the coordinates of the point are exactly on either of the upper limits of latitude or longitude, as indicated by the limit flag returned by subroutine FGRID, then the cell is actually outside the area covered by the Grid_Shift_Sub_File. To address this situation, the grid is virtually extended by one cell beyond the upper limit. Grid shift values are assigned to the corners of the cell, according to the limit flag, as follows:

[1] point is within the limits - shift values for all four corners of the cell are used, no virtual cell is required

[2] point is on the upper latitude limit - shift values for the two available cell corners are used for both the lower and upper latitude corners of the virtual cell

[3] point is on the upper longitude limit - shift values for the two available cell corners are used for both the lower and upper longitude corners of the virtual cell

[4] point is on both upper latitude and upper longitude limits - shift values for the single available cell corner are used for all four corners of the virtual cell

```
Case 1: (point is within limits)
~~~~~~
            d        c              d        c
             +------+                +------+
             |      |                |      |
             |  P • |   no change    |  P • |
             |      |                |      |
             +------+                +------+
            b        a              b        a


Case 2: (point is on upper latitude limit)
~~~~~~
                                    b        a
                                     +......+
                P                    :      :
             +---•--+    becomes     :      :
            b        a               :  P   :
                                     +---•--+
                                    b        a


Case 3: (point is on upper longitude limit)
~~~~~~
                 c                  c        c
                +                    +......+
                |                    :      |
              P •        becomes     :    P •
                |                    :      |
                +                    +......+
                 a                  a        a


Case 4: (point is on both upper latitude and longitude limits)
~~~~~~
                                    a        a
                                     +......+
              P                      :      :
                •        becomes     :      :
                 a                   :    P :
                                     +......•
                                    a        a
```

2. Interpolate Grid Shifts

The grid_shifts for the four corners of the cell containing the coordinates to be processed are interpolated using the bilinear interpolation technique.

```
 d               c        d     f       c
  +-------------+          +-----+-------+
  |             |          |     |       |
  |             |          |     |p      |
  |             |        h +-----•-------+ g
  |             |          |  {  |       |
  |             |          |  x  |~~~y~~~|
  |             |          |  {  |       |
  +-------------+          +-----+-------+
 b               a        b     e       a
```

Consider the unit square with shift values (a,b,c,d) at the corners, as illustrated above.
That is to say, the shifts are: a at (0,0), b at (1,0), c at (0,1), and d at (1,1).
The objective is to determine the shift p at (x,y), where 0<x<1, and 0<y<1.

To arrive at this unit square from the coordinate reference system of the grid, normalize as follows:

$$x = \frac{\text{latitude(p)} - \text{latitude(a)}}{\text{latitude\_interval}}$$

$$y = \frac{\text{longitude(p)} - \text{longitude(a)}}{\text{longitude\_interval}}$$

where

$$\text{latitude(a)} = (\text{row\_index} - 1) * \text{latitude\_interval}$$

$$\text{longitude(a)} = (\text{column\_index} - 1) * \text{longitude\_interval}$$

and thus

$$x = \frac{\text{latitude(p)}}{\text{latitude\_interval}} - (\text{row\_index} - 1)$$

$$y = \frac{\text{longitude(p)}}{\text{longitude\_interval}} - (\text{column\_index} - 1) \quad .$$

Next, determine the shift **e** at (0,y) by linear interpolation between **a** and **b**, and the shift **f** at (1,y) by linear interpolation between **c** and **d**.

$$e = a + (b - a) y$$

$$f = c + (d - c) y$$

Similarly, determine the shift **p** at (x,y) by linear interpolation between **e** and **f**.

$$p = e + (f - e) x$$

By substituting the expressions for **e** and **f** into the expression for **p**, we get

$$p = (a + (b - a) y) + ((c + (d - c) y) - (a + (b - a) y)) x$$

$$= a + (c - a) x + (b - a) y + (a - b - c + d) xy$$

The result would be identical if the first interpolations were computed for the shifts **g** at (x,0), and **h** at (x,1), and then the shift **p** at (x,y) interpolated between them.

Appendix A

Version 1 Grid_Shift_File

The Version 1 Grid_Shift_File consists of a Header_Block, a series of Grid_Shift_Records, and an End_of_File_Record (see Figure 3).

It is a FORTRAN 77 unformatted, direct-access file type, with a fixed record length of 16 bytes. Each record contains two data elements.

For the first 10 of the 12 Header_Records, the first data element is a Header_Element_Name which describes the data in the second data element. Header_Records 11 and 12 do not have descriptive Header_Element_Names.

There is a Grid_Shift_Record for each point in the grid. The two data elements are the shifts in the two directions of the grid. The number of Grid_Shift_Records is given by the calculation:

$$\text{Grid\_Points\_Per\_Row} = \frac{\text{Upper\_Longitude} - \text{Lower\_Longitude}}{\text{Longitude\_Grid\_Interval}} + 1$$

$$\text{Grid\_Points\_Per\_Column} = \frac{\text{Upper\_Latitude} - \text{Lower\_Latitude}}{\text{Latitude\_Grid\_Interval}} + 1$$

$$\text{Number\_of\_Grid\_Shift\_Records} = \text{Grid\_Points\_Per\_Row} * \text{Grid\_Points\_Per\_Column}$$

The last record in the Grid_Shift_File is an End_of_File_Record. This is used to ensure that no attempt is made to read beyond the end of data.

The following table specifies the contents of the Grid_Shift_File.

Record #	***First Data Element***	***Second Data Element***
Header Records		
1	'HEADER' Character*8	Number_of_Header_Records Integer*4
2	'S LAT ' Character*8	Lower_Latitude_Limit Real*8
3	'N LAT ' Character*8	Upper_Latitude_Limit Real*8
4	'E LONG ' Character*8	Lower_Longitude_Limit Real*8
5	'W LONG ' Character*8	Upper_Longitude_Limit Real*8
6	'N GRID ' Character*8	Latitude_Grid_Interval Real*8
7	'W GRID ' Character*8	Longitude_Grid_Interval Real*8
8	'TYPE ' Character*8	Grid_Shift_Data_Type Character*8
9	'FROM ' Character*8	System_From Character*8
10	'TO ' Character*8	System_To Character*8
11	Semi_Major_Axis_From Real*8	Semi_Minor_Axis_From Real*8
12	Semi_Major_Axis_To Real*8	Semi_Minor_Axis_To Real*8
Grid Shift Records		
13 ... ...n+12	Latitude_Shift Real*8	Longitude_Shift Real*8
End of File Record		
n+13	'END ' Character*8	End_Of_File Real*8

Appendix B

Version 2 Grid_Shift_File

The Version 2 Grid_Shift_File consists of an Overview_Header_Block, and one or more Grid_Shift_Sub_Files. Each Grid_Shift_Sub_File consists of a Sub_File_Header_Block and a series of Grid_Shift_Records. An End_of_File_Record follows the last Grid_Shift_Sub_File.

The Grid_Shift_File is a FORTRAN 77 unformatted, direct-access file type, with a fixed record length of 16 bytes. Each Header_Block_Record and the End_of_File_Record contain two data elements, while each Grid_Shift_Record contains four data elements.

For each Header_Block_Record, the first data element is a Header_Element_Name which describes the data in the second data element.

There is a Grid_Shift_Record for each point in the grid of each sub-file. The first two data elements are the shifts in the two directions (dimensions) of the grid. The next two data elements are the accuracies of the two shifts. The number of Grid_Shift_Records is given by the calculation:

$$\text{Grid\_Points\_Per\_Row} = \frac{\text{Upper\_Longitude} - \text{Lower\_Longitude}}{\text{Longitude\_Grid\_Interval}} + 1$$

$$\text{Grid\_Points\_Per\_Column} = \frac{\text{Upper\_Latitude} - \text{Lower\_Latitude}}{\text{Latitude\_Grid\_Interval}} + 1$$

$$\text{Number\_of\_Grid\_Shift\_Records} = \text{Grid\_Points\_Per\_Row} * \text{Grid\_Points\_Per\_Column}$$

After the last sub-file is an End_of_File_Record. This is used to ensure that no attempt is made to read beyond the end of data.

The following table specifies the contents of the Version 2 Grid_Shift_File.

Record #	***First Data Element***	***Second Data Element***
Overview Header Records		
1	'NUM_OREC' Character*8	Number_of_Overview_Header_Records Integer*4
2	'NUM_SREC' Character*8	Number_of_Sub_File_Header_Records Integer*4
3	'NUM_FILE' Character*8	Number_of_Grid_Shift_Sub_Files Integer*4
4	'GS_TYPE' Character*8	Grid_Shift_Data_Type Character*8
5	'VERSION' Character*8	Version_ID_of_Distortion_Model Character*8
6	'SYSTEM_F' Character*8	Reference_System_From Character*8
7	'SYSTEM_T' Character*8	Reference_System_To Character*8
8	'MAJOR_F' Character*8	Semi_Major_Axis_From Real*8
9	'MINOR_F' Character*8	Semi_Minor_Axis_From Real*8
10	'MAJOR_T' Character*8	Semi_Major_Axis_To Real*8
11	'MINOR_T' Character*8	Semi_Minor_Axis_To Real*8

Record #	***First Data Element***	***Second Data Element***

Sub-File Header Records

Record #	First Data Element	Second Data Element
1	'SUB_NAME' Character*8	Sub_File_Name Character*8
2	'PARENT' Character*8	Parent_File_Name Character*8
3	'CREATED' Character*8	Creation_Date Character*8
4	'UPDATED' Character*8	Last_Revision_Date Character*8
5	'S_LAT' Character*8	Lower_Latitude_Limit Real*8
6	'N_LAT' Character*8	Upper_Latitude_Limit Real*8
7	'E_LONG' Character*8	Lower_Longitude_Limit Real*8
8	'W_LONG' Character*8	Upper_Longitude_Limit Real*8
9	'LAT_INC' Character*8	Latitude_Grid_Interval Real*8
10	'LONG_INC' Character*8	Longitude_Grid_Interval Real*8
11	'GS_COUNT' Character*8	Number_of_Grid_Shift_Records Integer*4

Grid Shift Records

12 ... n+11 (NOTE: 4 elements per record)

Latitude_Shift	Longitude_Shift	Latitude_Accuracy	Longitude_Accuracy
Real*4	Real*4	Real*4	Real*4

End of File Record

Record #	First Data Element	Second Data Element
n+12	'END' Character*8	End_of_File Real*8