# NZGD2000 deformation model format

Version 1.0

Date 17 June 2013

## Introduction

This document describes the format used to publish the New Zealand Geodetic Datum 2000 (NZGD2000) deformation model. For more information about NZGD2000 see <http://www.linz.govt.nz/geodetic/datums-projections-heights/geodetic-datums/new-zealand-geodetic-datum-2000>

## Functional definition of the deformation model

The deformation model is defined as a component of the datum that is used to calculate the offset between the position of a mark at a given date (epoch) and the NZGD2000 reference coordinate, nominally its position on 1 January 2000. The offset can be determined at any time and location within the bounds of the model. Adding the offset to the reference coordinates gives the NZGD2000 epoch coordinate of the point at that time – for practical purposes the same as an ITRF96 coordinate.

At each location and time within its bounds the deformation model may define the following elements:

* horizontal displacement, represented as the east and north components of displacement in metres
* vertical displacement (upwards) in metres
* horizontal error, represented by the radius of the 95% confidence circle in metres
* vertical error, represented by the 95% confidence limit in metres.

The error model is simplistic, in that it does not model covariances between locations or allow different east and north errors.

The deformation model is built up from one or more submodels, representing different geophysical deformation sources. There is always a "national deformation model" (NDM) submodel, which represents the general tectonic deformation of the country. This may be supplemented by "patches", submodels representing the effect of specific earthquakes or other localised deformation events.

Each submodel is built of one or more components which when added together represent the total deformation due to the submodel. Submodels may define one or more of the four elements of the deformation at any time for which they are valid. Elements not defined by the component are assumed to be zero.

A component is built of a spatial representation which determines the value of the elements at any location within the bounds of the component, and a time function, which defines a time based scale factor that applies to the elements at any given time. Together they determine the values of the deformation for that component at the specified time and location.

The spatial representation may be defined by points on a regular grid aligned with latitude and longitude, or by a triangulated irregular network TIN, or by a nested group. A nested group is an ordered list of grid and/or TIN representations, from which the first one that applies at a location defines the values of the elements at that location.

The time function can be one of a velocity, step, ramp, or exponential decay.

The details of calculating deformation elements for each component and for combining them to give the total deformation are described below. From the point of view of calculating deformation elements, the assignment of components to submodels is irrelevant – each component is calculated individually and combined as described below.

The NZGD2000 deformation is periodically updated with new versions, for example after earthquakes, or when the NDM or a patch has been recalculated. Versions are numbered according to the date on which they are defined. The published deformation model includes information for all previous versions as well as the current version. The model assigns each component an initial version in which it first applies, and optionally a version in which it is revoked (typically if it is replaced by a revised component). These attributes are used to determine which components are applicable to each version of the model.

Note that currently the NZGD2000 model does not require using a TIN spatial representation. While a TIN may be desirable to represent complex deformation close to a fault, it is much more complex (and computationally slow) to calculate on a triangulated mesh, so a multi-resolution grid should be used in preference. The TIN option is catered for in the model specification as it is possible that it will be required in the future.

Similarly the current model does not define error elements for the deformation model, or use exponential decay time functions.

### Gridded spatial representation

Gridded spatial representations are defined as regular grids in terms of latitudes and longitudes. That is, longitude (x) and latitude (y) of a grid node is defined as

xi = xo + i.dx

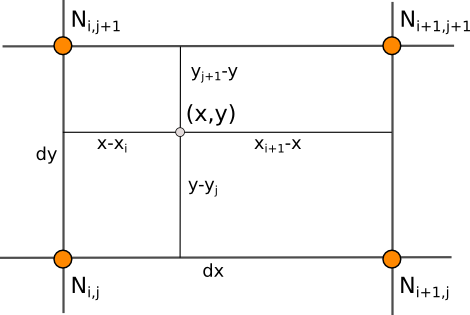
yj = yo + j.dy

where xo, yo are the longitude and latitude of the southwest-most corner of the grid, dx and dy are the longitude and latitude grid spacing, and i and j are the column and row number of the grid cell (where the west-most column and southernmost row are numbered 0). Note that the longitude and latitude grid spacing need not be equal – it is preferred that dx is approximately equal to dy/cos(ym), where ym is the latitude of the middle of the grid, as this makes the grid cells approximately square (except at polar latitudes).

Displacement vector elements are calculated using bilinear interpolation with respect to latitude and longitude from the nodes at the corners of the grid cell within which the calculation point lies. Each element of the displacement is calculated independently (though of course the interpolation weighting will be the same for each, as they all refer to the same calculation point).

Bilinear interpolation is defined as follows:

The calculation point (x,y) is located in the grid cell between columns i and i+1, and rows j and j+1.

The displacement elements (de, dn, du) at the calculation point are weighted means of the corresponding elements at the four nodes.

The weights are calculated as follows:

Wi,j = ((xi+1-x)/dx) \* ((yj+1-y)/dy)

Wi+1,j = ((x-xi)/dx) \* ((yj+1-y)/dy)

Wi,j+1 = ((xi+1-x)/dx) \* ((y-yj)/dy)

Wi+1,j+1 = ((x-xi)/dx)\*((y-yj)/dy)

So for example the east displacement at the point (x,y) is calculated as

de = Wi,j\*dei,j + Wi+1,j\*dei+1,j + Wi,j+1\*dei,j+1 + Wi+1,j+1\*dei+1,j+1

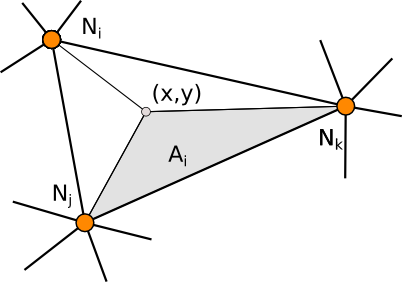
The error elements eh, ev are interpolated using a weighted average of the variances eh2, ev2, for example

eh = √(Wi,j\*ehi,j2 + Wi+1,j\*ehi+1,j2 + Wi,j+1\*ehi,j+12 + Wi+1,j+1\*ehi+1,j+12)

### Triangulated irregular network spatial representation

For TIN spatial representations the triangulation is defined by a set of nodes at which the displacement is defined. Each node is assigned an integer id. The triangulation also defines a set of triangles, each defined by a sequence of three node ids ordered in an anticlockwise direction (when viewed from above the triangle). The triangulation is required to be defined such that it forms a consistent triangular mesh with a single outer boundary, no holes, no overlaps, and such that the outer boundary is a convex polygon in terms of latitude and longitude coordinates.

Displacement vector elements are calculated using linear interpolation over the triangle in terms of longitude and latitude coordinates. Each component of the displacement is interpolated independently.

The interpolation is defined as follows:

The calculation point (x,y) is located in a triangle with nodes Ni, Nj, Nk at coordinates (xi,yi), (xj,yj), (xk,yk).

A weight factor W is calculated for each node, which is equal to the ratio of the area of the triangle formed by the calculation point and the other two nodes to the total area of the triangle. For example for node Ni this is the ratio of area Ai to that of the triangle Ni, Nj, Nk. This evaluates to:

Wi = ((x-xj)\*(yk-yj) - (y-yj)\*(xk-xj))/((xi-xj)\*(yk-yj) - (yi-yj)\*(xk-xj))

Wj = ((x-xk)\*(yi-yk) - (y-yk)\*(xi-xk))/((xj-xk)\*(yi-yk) - (yj-yk)\*(xi-xk))

Wk = ((x-xi)\*(yj-yi) - (y-yi)\*(xj-xi))/((xk-xi)\*(yj-yi) - (yk-yi)\*(xj-xi))

The displacement elements (de, dn, du) and errors (eh, ev) at the calculation point are then calculated using weighted averages of the values at the nodes in the same way as described above for grids.

### Time functions

The deformation model supports four time functions. Each may be used to calculate two scale factors that apply at any given time t. These are ft which applies to the displacement elements of the component de, dn and du and fe,t which applies to the error elements eh and ev.

The time functions are defined in terms five parameters – start time t0, end time t1, start factor f0, end factor f1, and decay rate ε. Times are expressed as dates with or without times. Factors are undimensioned multiplication factors. Decay rate is the exponential decay rate expressed in years (ie representing the time taken for the remaining post-seismic displacement to reduce by a factor of e (=2.718...).

The time functions are defined as follows:

|  |  |  |
| --- | --- | --- |
| linear velocity | ft = (t – t0) | Currently used for the deformation model |
| linear ramp | ft = f0 for t < t0  ft = (f0.(t1 – t) + f1.(t-t0))/(t1-t0)   for t0 <= t < t1  ft = f1 for t >= t1 | Used for piecewise linear representation of, for example, post-seismic deformation |
| step function | ft = f0 when t < t0,  ft = f1 when t >= t0 | A special case of the linear ramp when t0 = t1. Used for conventional patch (f0 = 0, f1 = 1), or reverse patch (f0 = -1, f1 = 0) |
| exponential decay | ft = f0 for t < t0  ft = f0 + (f1-f0)(1-eε (t0-t))/(1- eε (t0-t1))  for t0 <= t < t1  ft = f1 for t >= t1 | This is used to represent deformation resulting from post-seismic deformation |

For the error elements eh, ev the scale factor fe,t is defined by:

|  |  |  |
| --- | --- | --- |
| Velocity function | fe,t = ft | Note this may be negative. However it is squared when it is used. |
| Step, ramp, and exponential function | fe,t = √abs(ft) |  |

This approach is used because the ramp, step, and exponential decay models typically have scale factors less than one, and the ramp and step models may be combined in a piecewise linear model. As error elements are added using the root sum of squares, the √f factor gives a more sensible error.

Where it is necessary to convert a number of days to a number of years, a year is defined as 365.2425 days.

### Combination of components

At a given time and location the elements from each component of each submodel are combined to determine the overall displacement and errors.

The displacement elements de, dn, dh are combined by simply adding their values calculated for each component. For example, if there are n components for which the spatial representation calculates de as de1, de2, … to den, and the time function evaluates to f1, f2, … to fn then the total model value for de is

de = f1.de1 + f2.de2 + … + fn.den

The error values eh, ev are combined by determining the root sum of squares (RSS) of the values determined for each component. So for example

eh = √(fe,12.eh12+ fe,22.eh22+ … + fe,n2.ehn2)

### Calculation of deformation between two epochs

Calculating the deformation between two times is straightforward for the displacement elements de, dn, and du as it is simply the difference between the values calculated at each time.

This approach is not appropriate for the error components eh, ev. Uncorrelated errors are combined as a root sum of squares, but the errors of displacements calculated for one component calculated at different times are clearly correlated.

While there is no mathematically correct way to define the errors without a much more complex error model, the following approach is recommended if these errors are required.

The time function error factor of the difference between t0 and t1 is calculated for each component separately as fe,t1-t0. For step, ramp, and decay functions this is calculated as

fe,t1-t0 = √abs(fe,t12-fe,t02)

For the velocity function this is calculated as

fe,t1-t0 = fe,t1 – fe,t0

The eh and ev values from the spatial representation of each component are multiplied by these time function error factor values and then combined as the root sum of squares to give the total error of the deformation between the two epochs.

## Distribution format

The model is distributed as a set of CSV (comma separated files) in a well defined directory structure, embedded in a zip file. Additionally it includes ANZLIC compliant metadata descriptions (<http://www.linz.govt.nz/geospatial-office/about/projects-and-news/anzlic-metadata-profile>), as well as documentation describing the format, version information, other reference information (eg about the models used), and optionally tools to process, test, or reformat the model.

The model will be versioned, with each release having a version number corresponding to the release date (eg 20130801). The version number does not imply anything about what deformation events are included in the model. The zip file will be named according to the latest version it contains, for example nzgd2000\_deformation\_20130801\_full.zip.

The zip file root directory will contain subdirectories *model*, *documentation*, and optionally *tools* and *test.*

The *model* directory contains the data defining the actual model. Within it will be a directory *ndm* containing the national (secular) model, and a directory for each patch. The model directory will contain a file called *model.csv* defining the submodels of the model, a file called *versions.csv* with a table of version history, and a file called *metadata.xml* containing the ANZLIC metadata describing this data set. Some of the metadata is replicated in a CSV file *metadata.csv*. The *model.csv* will define each submodel. Within each submodel directory is a file *component.csv* specifying the names of the file(s) defining the grid or TIN displacement field, the spatial and temporal extents over which the patch applies, and the model versions for which it applies. The component subdirectory also contains the grid or triangulation files.

### Model specifications

* The model is released periodically. Each release is assigned a version number encoding the date of the release in format YYYYMMDD (year, month, day).
* The model comprises a “national deformation model” (NDM) defining the secular tectonic deformation of New Zealand and an arbitrary number of patches defining the deformation from specific events. Each of these is termed a submodel.
* The NDM and each of the patches may comprise a number of components used to represent the distribution of deformation in time and space
* Each component can define any of the following elements – horizontal displacement (de, dn), vertical displacement (du), horizontal error (eh), vertical error (ev). Horizontal and vertical are defined in terms of ellipsoidal coordinates (that is the north and east directions are aligned with true north and east, not grid north and east on a projection).
* The displacement field calculated at any given time is spatially continuous – any patch that does not cover the full extent of the deformation model will calculate to zero at its edges. (Note that if modelling close to a fault rupture the actual displacement will be discontinuous. This will not be properly reflected in the model – in order to ensure that the deformation is invertible it will include an arbitrary smoothing across areas of surface rupture.)
* The deformation at any given time and place is determined by summing the values calculated for each component of each submodel (NDM and patches) at that time and place. The calculation is based on the latitude and longitude of the point ignoring elevation.
* A component can evaluate as undefined at a given time and location. If any component evaluates as undefined, then the entire deformation model is undefined at that time and location. (Note that while patch models do not explicitly cover the extent of the deformation model, they are specified as evaluating to 0 outside their extents, not evaluating to undefined). For the New Zealand model this may apply to parts of the grid that are offshore, as well as points beyond the extents of the NDM. There is no specifically defined behaviour where the deformation is undefined – it is implementation dependent.
* The displacement field calculated at any given time will be uniquely invertible. That is, no two locations will transform to the same point after applying the displacement field. This is required to be able to transform locations uniquely between any two epochs. Note that this is a constraint on how precisely faulting can be represented. (For velocity models this will only be true over "reasonable" time frames)
* Each component defines a spatial representation and time function. The spatial representation defines a displacement vector and/or error components at any location within the range of the model, and the time function defines a scale factor at any time within the valid date range of the component.
* The spatial model is defined either by values on a rectangular grid in terms of latitude and longitude, or by a TIN in terms of latitude and longitude or by a prioritised group of grid and/or TIN definitions. The displacement field is defined at each node of the grid or mesh, and interpolated by bilinear interpolation (grids) or linear interpolation (TIN). (Details below)
* Where the spatial representation is defined by a grid, TIN, or nested group. The value of a nested group at any location is the value of the first TIN or grid in the group which is defined at that location (typically this would apply for nested grid definitions, where the finest grid applying at a location would be used to calculate the deformation).
* Several components may use the same grid or TIN spatial representation with different time functions.
* The time function can use one of four time functions, “velocity”, “step”, “ramp”, “decay”.
* Horizontal deformation at each node is defined in terms of metres (or for velocities, metres/year) in the east and north direction as defined at the latitude and longitude of the node on the ellipsoid (ie at ellipsoidal height 0). (While it would be simpler to define the horizontal components in terms of degrees east and north this is less intuitive for reading – an implementation may apply this conversion).
* Vertical deformation at each node is defined in terms of metres upwards.
* Horizontal and vertical errors represent the 95% circular confidence and 95% confidnce level of the deformation at each node in metres.

### Data specifications

* All coordinates are supplied as NZGD2000 longitude and latitude (note: in principle this creates an issue when reverse patches are applied, as technically the coordinates change. However in practice this will not make any difference unless we create models that fit within a few metres of a fault trace – not currently proposed). Longitudes and latitudes are expressed as decimal degrees.
* All displacements are defined in terms of metres, and velocities in metres/year. These are defined (optionally) for the east, north, and up displacements, where these directions are in terms of the datum ellipsoidal coordinate system.
* Dates are specified in CSV files using the format YYYY-MM-DD or YYYY-MM-DD hh:mm:ss. Date/times are assumed to be UTC (where it matters)

### File specifications

* Each version of the model is released in two single zip files. The file nzgd2000\_deformation\_*yyyymmdd*.zip contains the information required to calculate the deformation for the version yyyymmdd, and the file *nzgd2000\_deformation\_yyyymmdd\_full.zip* contain the information required to calculate the deformation for version yyyymmdd and all previous releases of the model.
* The zip file base directory contains subdirectories *model*, *documentation*, and optionally *tools* and *test*.
* The zip file base directory contains a file VERSION, containing a single line of text with just the version number
* The model directory provides all the information required to calculate any released version of the deformation model at any location and time for which it is defined
* The model directory contains a file *metadata.xml* which contains the model metadata following the ANZLIC metadata standard (ref)
* The model directory contains a free format text file *metadata.csv* holding some basic metadata.
* The model directory contains a directory called *ndm* containing the national deformation model data
* The model directory contains a directory for each patch containing the data for that patch. The directory is called *patch\_xxxxx* where *xxxxx* is an arbitrary text string (typically representing the deformation event, eg chch201112)
* The model directory contains a CSV format file called *model.csv* which defines all the submodels (ndm and patches) of the current and previous versions of the deformation model
* The ndm and each patch directory contains a file called *component.csv,* which defines the components of the current and previous versions of the NDM or patch.
* Grid data is stored in CSV files named *grid\_xxxx.csv* – one for each grid.
* TIN data is stored in CSV files name *trig\_pts\_xxxxx.csv* defining the nodes in the triangulation, and *trig\_trg\_xxxxx.csv* defining how triangles are formed between the nodes
* Documentation is supplied as either PDF formatted files, or HTML encoded text (plus stylesheets, PNG images, etc)
* All file and directory names are case sensitive. For example the model.csv file is not included as Model.csv. The case of the ndm and patch directories have the same case as specified in the model.csv file, and the grid and trig files have the same case as specified in the component.csv files.
* All files and directory names differ by more than case (ie file names are not ambiguous on a case insensitive file system)
* File and directory names comprise only the characters a-z, numbers 0-9, and the underscore and point characters (they do not include spaces)
* All text files use UTF-8 encoding and are not prefixed with a byte order marker (BOM).
* The model.csv, component.csv, and grid and TIN csv file contents are restricted to the ASCII character set (ie UTF-8 is equivalent to ASCII for these files)
* The CSV format follows excel conventions. That is fields are delimited by a single comma character. Any field containing a comma, double quote, carriage return, line feed or other non-ASCII character must be delimited with double quotes ("). Double quotes within quoted fields are represented by a pair of double quote characters. End of record is delimited by CR/LF (ie windows format). The presence of quotes does not imply data type – that is numeric fields may be quoted, and text fields may be unquoted (unless their content requires quoting)
* The first row of each CSV file contains the names of the fields in the file. The required fields for each csv file type are listed below. Fields will be in the order listed – reading the header to determine the fields is optional but recommended. Grid and triangulation CSV files included optional fields, depending on the components of deformation that they define. Optional fields will be omitted if not required.

### metadata.csv specification

The metadata.csv file contains some simple metadata about the model. It contains the following fields:

|  |  |
| --- | --- |
| item | The name of the metadata item |
| value | The value of the item |

The file will contain at least the following items (which may be blank):

|  |  |
| --- | --- |
| model\_name | A name for the model |
| description | A description of the model |
| version | The current version of the model |
| datum\_code | A code for the datum for which the model applies (NZGD2000) |
| datum\_name | The name of the datum |
| datum\_epoch | The reference date for the datum coordinates |
| datum\_epsg\_srid | The spatial reference id of the datum latitude/longitude coordinate system |
| ellipsoid\_a | The length of the semi-major axis of the ellipsoid in metres, used for converting north/east displacements to latitude/longitude displacements |
| ellipsoid\_rf | The reciprocal of the flattening of the ellipsoid, used for converting north/east displacements to latitude/longitude displacements |
| authority | The name of the authority sourcing the deformation model |
| authority\_website | URL of the authority website |
| authority\_address | Contact address for the authority |
| authority\_email | Contact email for the authority |
| source\_url | Url for downloading the latest version of the deformation model |

### version.csv specification

The version.csv file provides a brief history of the versions of the model used. It contains the following fields:

|  |  |
| --- | --- |
| version | The version number of the version described |
| release\_date | The release date of the version (should match the version) |
| reverse\_patch | 'Y' if the version includes a new reverse patch – meaning that “reference coordinates” should change. The required reference coordinate change can be determined from the model data (by extracting components added or revoked in the version which have a non-zero displacement at the reference epoch) |
| reason | Text description of the reason for releasing the version |

### model.csv specifications

The model.csv file defines the submodels of the model and contains the following fields in the order specified.

|  |  |
| --- | --- |
| submodel | The name of the submodel directory (ndm or patch name) |
| version\_added | The version of the deformation model in which this component was first defined |
| version\_revoked | The version of the deformation model in which the component was revoked – 0 if it still applies to the current version |
| reverse\_patch | 'Y' if the submodel implements a reverse patch (requires a change to reference coordinates). |
| description | Free text description of the submodel |

### component.csv specifications

The component.csv files defines components of the NDM or patch. Components which have a grid or TIN spatial representation are represented by a single record in the file. Components using a nested group for spatial representation have a record for each grid or TIN file in the group. The field component in the file identifies the component to which the record refers. All records relating to a nested group component have the same component id.

The component.csv file includes the following fields in the order specified. See the data specifications above for units and format of entries

|  |  |
| --- | --- |
| version\_added | The version of the deformation model in which this component was first defined |
| version\_revoked | The version of the deformation model in which the component was revoked – 0 if it still applies to the current version |
| reverse\_patch | 'Y' if the component implements a reverse patch (ie requires a change to reference coordinates). |
| component | Positive integer id of the nested group to which this record belongs. value of 0 indicates that this item is a complete grid or TIN component. |
| priority | Positive or 0 integer id of the priority of the item. Only applies to nested groups. To calculate the deformation for the component at a location highest priority subcomponent that is defined at that location is used. |
| min\_lon | The minimum longitude value of the points defining the grid or TIN |
| max\_lon | The maximum longitude value of the points defining the grid or TIN |
| min\_lat | The minimum latitude value of the points defining the grid or TIN |
| max\_lat | The maximum latitude value of the points defining the grid or TIN |
| spatial\_complete | “Y” if the component is 0 outside the longitude/latitude range, “N” if it is undefined outside this range (typically this will be N for the NDM, Y for patches) |
| min\_date | The earliest date for which the time function is non-zero, or 0 if is unbounded |
| max\_date | The latest date for which the time function is non-zero, or 0 if it is unbounded |
| time\_complete | “Y” if the component is 0 outside the date range, “N” if it is undefined |
| npoints1 | The number of columns (longitude values) of a grid, or the number of nodes in a TIN |
| npoints2 | The number of rows (latitude values) of a grid, or the number of triangles in a TIN |
| displacement\_type | One of “horizontal”, “vertical”, “3d”, or "none". |
| error\_type | One of “horizontal”, “vertical”, “3d”, or "none". |
| max\_displacement | The maximum length of a displacement vector (using the displacement vectors defined at the grid or triangulation nodes before scaling to account for the time function) |
| spatial\_model | One of “llgrid” (longitude/latitude grid) or “lltin” (TIN) |
| time\_function | One of “velocity”, “step”, “ramp”, “decay” |
| time0 | The t0 value of the time function |
| factor0 | The f0 value of the time function |
| time1 | The t1 value of the time function |
| factor1 | The f1 value of the time function |
| decay | The exponential decay rate for post-seismic movement |
| file1 | The name of the grid file or triangulation point (relative to the directory holding the component.csv file) |
| file2 | The name of the triangulation triangle file (blank for grid models) |
| description | Free text description of the model component |

Notes:

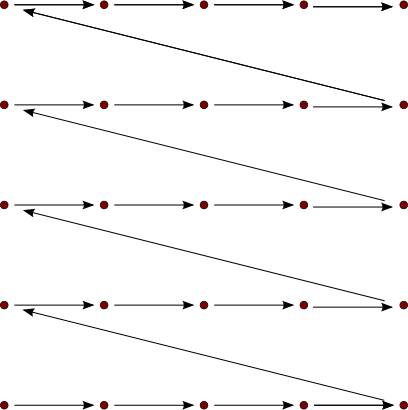
* the version\_added and version\_revoked fields determine whether the component applies to a specific version of the model. The current model comprises all components for which the revoked\_version is 0. To calculate a specific version use all the components for which version\_added <= version < version\_revoked.
* Rows with the same non-zero component id (ie rows that comprise a nested spatial definition) can only differ in the priority, min\_lon, min\_lat, max\_lon, max\_lat, npoints1, npoints2, file1, file2 and description. The priority must be unique for each row with the same non-zero component id.
* the reverse\_patch flag identifies whether the time function is zero at the coordinate reference epoch. If it is non zero then the reverse\_patch flag is Y, otherwise N.
* the minimum and maximum values of longitude, latitude, and date are used to determine components which cannot be calculated at given time and location. The result of calculating the component at a time and location is either 0 or undefined, depending upon the values of spatial\_complete and temporal\_complete.
* If spatial\_complete is N then the component is undefined outside the the latitude/longitude range (or for TIN models the extents of the TIN within this range). Otherwise it evaluates to a zero displacement vector.
* If temporal\_complete is N then the time function is undefined outside the date range of the model. Otherwise it evaluates to a zero scale factor outside this range.
* the values npoints1 and npoints2 are to simplify reading grid and TIN models, allowing storage to be allocated before the model is read. For a grid model they can also be used with the min and max longitude and latitude to calculate all the grid node coordinates (the values in the grid CSV file are redundant).
* several rows (both current and historical) may refer to the same grid or TIN files
* "displacement\_type" and "error\_type" cannot both be "none".

### Grid representation csv specifications

The grid representation CSV files contain the following fields in this order:

|  |  |
| --- | --- |
| lon | The longitude of the point |
| lat | The latitude of the point |
| de | The east displacement/velocity (present if displacement\_type is horizontal or 3d) |
| dn | The north displacement/velocity (present if displacement\_type is horizontal or 3d) |
| du | The vertical displacement/velocity (present if displacement\_type is vertical or 3d) |
| eh | The horizontal error at the point in metres (present if error\_type is horizontal or 3d) |
| ev | The horizontal error at the point in metres (present if error\_type is vertical or 3d) |

Note:

* the grid nodes are entered in the CSV file in the order illustrated, starting at the southernmost row, ordering the values of the row from west to east, repeating for each row and finishing at the NE corner of the grid.
* The displacement components may be undefined at a node. Undefined values are represented by a blank (zero length string) entry

### Triangulated irregular network representation csv specifications

The TIN representation is defined by two files, the points file *trig\_pts\_xxxxx.csv* defining the nodes in the triangulation, and triangulation file *trig\_trg\_xxxxx.csv* defining how triangles are formed between the nodes.

The first defines the triangulation nodes with the following fields

|  |  |
| --- | --- |
| id | The id used to reference the point. This is an integer, increasing sequentially from 1 in the first line of the file (after the header), to npoints1 in the last line. |
| lon | The longitude of the point |
| lat | The latitude of the point |
| de | The east displacement/velocity (present if displacement\_type is horizontal or 3d) |
| dn | The north displacement/velocity (present if displacement\_type is horizontal or 3d) |
| du | The vertical displacement/velocity (present if displacement\_type is vertical or 3d) |
| eh | The horizontal error at the point in metres (present if error\_type is horizontal or 3d) |
| ev | The horizontal error at the point in metres (present if error\_type is vertical or 3d) |

The displacement components may be undefined at a node. Undefined values are represented by a blank (zero length string) entry.

The second file defines how the triangles are constructed on the mesh, and contains the following fields

|  |  |
| --- | --- |
| id1 | The first node in the triangle |
| id2 | The second node in the triangle |
| id3 | The third node in the triangle |

Note:

* the order of triangles in the file is arbitrary, but the nodes of a triangle must be ordered such that traversing from point id1 to point id2 to point id3 defines an anticlockwise ordering around the triangle when viewed from above.