Land Information New Zealand
Toitū te whenua

Proposal for encoding of a Deformation Model

Chris Crook <ccrook@linz.govt.nz>
Land Information New Zealand
4 December 2019

Overview

This is a proposal to encode a time based transformation function, typically a deformation model, into a binary formatted data file that can be used directly by transformation software. These deformation models are typically developed by national geodetic agencies to be used by geospatial and positioning communities to transform between national reference frames and global reference frames. The audience for this proposal are geodetic agencies publishing deformation models, and developers of geospatial and positioning software that consumes them.

Currently there is no common format for such models, and typically agencies develop both the model and software to evaluate it at any given time and location. The software may not interface well with consumer software where the transformation may be required, such as in GIS (Geographic Information Systems) and positioning software. The formats are built to serve the needs of the individual agency, generally using custom binary formats with limited metadata.

This proposal describes a format that can be used to represent most deformation models currently in use, and that contains sufficient metadata to clearly describe the model and its intended usage. The format is intended to support efficient coordinate transformations directly by software without further preprocessing. It builds on well defined existing formats to simplify implementation into software stacks.

The proposal is for a multiple file format comprising one or more nested grids of displacements accompanied by a JSON file defining the time behaviour associated with each grid. The JSON file also contains metadata of the model.

The binary formatted nested grids are based on the GeoTIFF format as proposed by in the PROJ project request for comments RFC 4 (PROJ/rfc-4.rst at rfc4_remote_and_geotiff_grid · rouault/PROJ). This only requires extending the types of grids that can be encoded to potentially encode horizontal and vertical uncertainty.

This proposal is offered as a straw man with the hope that it will draw support from potential publishers and consumers of deformation models. After it has been adapted based on feedback from potential users of the format it is intended to implement it into the PROJ library which underpins coordinate transformations in most of the open source geospatial stack. This will provide a working model and tools for building and validating deformation models. Ideally this could then serve as a basis for defining a standard for a deformation model format.

Acknowledgements

I am very grateful to suggestions from numerous contributors that have greatly improved this proposal. In particular Kristian Evers in relation to algorithms for deformation the current PROJ +deformation method, and Even Rouault for the GeoTIFF format and many recommendations that have improved the content of the master file. Finally thank you to everyone who has commented on the proposal or the very useful related discussions around implementation and use of deformation models in coordinate transformations. It has all helped enormously.

Related work

This functional model is based on that developed by Land Information New Zealand in 2013 to encode and publish the NZGD2000 deformation model (https://www.linz.govt.nz/data/geodetic-system/datums-projections-and-heights/geodetic-datums/new-zealand-geodetic-datum-2000-nzgd2000/nzgd2000-deformation-model. The principal difference between this model and the LINZ model is that the LINZ format is an ASCII format intended purely for publication, and is not efficient for calculation. The LINZ model less clean in that it is based on individual grids rather than an underlying nested grid structure.

The GeoTIFF grid format proposal for PROJ (https://github.com/rouault/PROJ/blob/rfc4_remote_and_geotiff_grid/docs/source/community/rfc/rfc-4.rst) has triggered this proposal. This format provides a base capability for encoding nested 3 dimensional gridded deformation. It could be extended to fully encode a time dependent deformation transformation.

This was a similar enhancement request raised on the PROJ project in 2018 to develop a deformation model format (https://github.com/OSGeo/PROJ/issues/1001). After much discussion that stalled as there were no clear candidate formats. The creation GeoTIFF has provided the missing capability for building a simple binary deformation model format.

The ESRI GGXF initiative (https://github.com/Esri/ggxf) is another development that may provide a suitable format for encoding a deformation model. After considering many options for a generic grid format for geodetic data it has settled on NetCDF/HDF5 (https://www.unidata.ucar.edu/software/netcdf/, https://support.hdfgroup.org/HDF5/whatishdf5.html) as a base format. Similarly to the TIFF grid format proposal this would need extending with a schema/profile to provide a time dependent transformation function format.

Most of the following proposal is agnostic encoding in either TIFF or GGXF formats

Motivation

Accurate transformation of geospatial data between time dependent datums commonly requires using a deformation model. The deformation model approximates the movement of the earth's surface and the movement of features on it that we intuitively think of as "fixed".

This deformation can be used either as a transformation to a "plate-fixed" coordinate reference system, such as the NZGD2000 coordinate system used in New Zealand, or as a point motion model to determine the location of a feature at a datum reference epoch.

Within stable continental plates this movement may be emulated to an adequate level of accuracy by a simple plate rotation model – a constant rate of rotation about an axis through the centre of the earth and extending through the the surface of the surface of the earth at the Euler Pole of the rotation. This can be simply defined by defined by four parameters, though commonly encoded in a 14 parameter Bursa-Wolf transformation with a defined reference epoch.

However many areas are subject to more complex deformation, including large scale secular crustal deformation near plate boundaries and vertical deformation due to Glacial Isostatic Adjustment (GIA). These can often be represented by a constant velocity model. Additionally

many areas suffer episodic deformation events such as earthquakes and aseismic slow slip event.

In order to transform coordinates of points affected by such movements a model of the deformation must be encoded into the transformation software.

Until recently such deformation models have not been supported other than in specialized software of very limited application.

The PROJ coordinate transformation library has now provided a capability to use a grid model as a velocity deformation field, as well as creating transformation pipelines that can combine multiple transformation steps to emulate secular deformation and episodic events.

While this is a great step forward it does not fully support specifying a complex deformation model in which the selection of components to apply will depend on the transformation date. Furthermore it does not provide a generic deformation model format that can be implemented in other software.

There is currently no way to provide to publish a transformation model in a portable binary format that can be used by a range of software. There is not even a nested grid format suitable for representing three dimensional deformation such as that caused by an earthquake. Representing deformation effectively using grids often requires a hierarchy nested gridswith finer grids nearer the epicentre where the deformation is complex, and coarser grids to represent the relatively smooth far field deformation.

The PROJ software is currently working on a new grid format based on the TIFF specification (originally designed for image data). This will provide a capability for encoding a nested grid structure with 1, 2, 3, or more values defined at each grid node. This provides a format suitable for defining the components of a deformation model, such as the secular velocity field, or the deformation due to a specific episodic event.

This document proposes extending this to fully encode a deformation model. This may include multiple components such as secular deformation, earthquake co-seismic and post seismic deformation, and glacial isostatic adjustment (GIA), each with a defined time behaviour. The intention is to provide a format sufficiently generic to support publishing most, if not all, deformation models currently in use in a format that is directly and efficiently usable by transformation software.

While this initially motivated by the opportunity presented by the TIFF encoding of gridded data much of this proposal is agnostic to format. Any grid format will suffice if it can encode nested grids with an arbitrary number of data values per node could serve the purpose.

Role of the deformation model

The deformation model is simply a function that defines the movement of a region of the earth's crust and objects fixed upon it. It is more accurately termed a movement model or displacement model. Deformation is a consequence of that movement. The term comes from geophysics where this movement is studied to measure crustal deformation and understand the stresses that it induces.

In the context of coordinate systems a deformation models can serve either of two roles:

- as a point motion model defining the position of objects in a coordinate system relative to their points at a reference epoch
- as a time dependent coordinate transformation between two coordinate systems.

The manner of defining the model and the calculations required to evaluate the model are practically identical in each case. They differ in that:

- the point motion model is defined in terms of a single coordinate system and a reference epoch, whereas the time dependent transformation is defined in terms of a source coordinate system and a target coordinate system
- the point motion model evaluates to zero at the reference epoch, whereas the time dependent transformation may not have an epoch at which it evaluates to zero.

Mathematically the deformation model is a function which evaluates an east, north, and up offset as a function of latitude, longitude, and time (or easting, northing and time if it is based on a projection coordinate system).

In a point motion model the offset is added to the coordinate of a "fixed" object at the reference epoch to estimate its position at the evaluation epoch.

In the time dependent transformation the offset is added to the source coordinate of an object at the evaluation epoch to calculate the corresponding coordinate in the target coordinate system at that epoch.

Functional definition of the deformation model

The functional model proposed for a general deformation model can be characterised as follows:

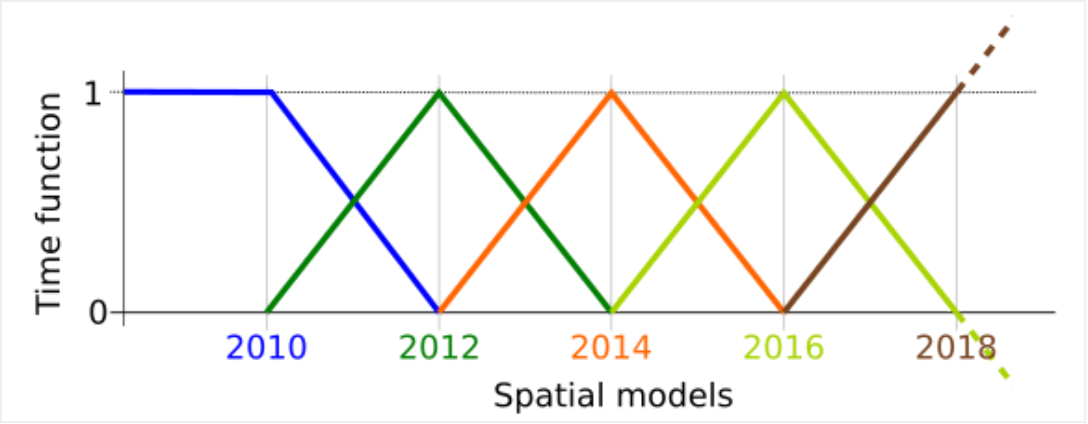
- the total deformation at any time and location is defined by summing the deformation of multiple components. Typically each component describes an aspect of the deformation, such as glacial isostatic adjustment, secular deformation, earthquakes.
- each component may define any of horizontal displacement, vertical displacement, horizontal uncertainty, and vertical uncertainty
- each component is defined by a time function and a spatial model. The component deformation at a given time and place is determined by multiplying the movement calculated from the spatial model at that location by a scale factor determined from the time function
- each spatial model is defined by one or more grids forming a nested grid model. The deformation is interpolated using bilinear interpolation from the deformation at each grid node.
- spatial models may define deformation on a subset of the full extent of the deformation model. For example a model defining earthquake deformation may have limited extent. Where the spatial model is defined over a subset of the extents of the deformation model it is assumed to be zero beyond that subset.
- each time function is a simple mathematical function of time which evaluates a scalar scale factor applied to the spatial model. Initially two time functions are supported – a piecewise linear function of time in which the scale factor is defined at a sequence of times, and an exponential function (which is a useful approximation for some post-seismic behaviour).

Discussion: This is intended as a minimal model to represent a deformation field – a "minimal viable product". In the future it could be extended to incorporate different spatial models, interpolation methods, and time functions.

This proposal assumes that the deformation model is represented using a multiple grid representation. Currently most (if not all) deformation models in use are gridded models so this seems a sensible restriction of the proposal.

An example of a special case that this model can easily encode is a time series of spatial models, effectively a three dimensional grid with dimensions longitude, latitude, and time. In this case each component grid has a "double ramp" time function running from 0 at the time of the previous grid up to 1 at the time of the current grid, and then down to 0 at the time of the next grid. (The time functions for the final to grids could be extended linearly to allow the deformation to be interpolated until the next grid becomes available.) This is illustrated

below.



In the future there may be value in using some other representation than nested grids for the deformation model. For example structures such as Discrete Global Grid Systems provide a global grid of varying level of detail. As these acquire more support in software and if there is a drive to develop a global deformation model then this may be worth developing.

Triangulated models may be more efficient to represent complex deformation near surface faulting. However this may not be desirable for defining or relating to a geospatial datum. In New Zealand triangulated models were considered for modelling the deformation due to the 2011/12 Christchurch earthquakes but didn't offer much advantage in the size of the model, and also are much less efficient to evaluate since it is necessary to search the triangulation to evaluate at any particular location. (See https://www.linz.govt.nz/system/files_force/media/file-attachments/winefield-crook-beavan-application-localised-deformation-model-after-earthquake.pdf?download=1).

The following sections detail the attributes required to represent this functional model.

Deformation parameterisation

Each deformation model component can define any or all of the following attributes at a location to characterise the deformation:

- horizontal displacement – the east and north displacement in metres or degrees
- horizontal uncertainty – the circular 95% confidence limit in metres
- vertical displacement – the upwards vertical displacement in metres
- vertical uncertainty – the 95% confidence limit in metres

The uncertainties can either be specified in the spatial model and interpolated in the same way as displacements or as constant values for the entire component.

The vertical direction will be the local vertical of the coordinate system. For deformation there is no practical need to distinguish gravitational and geometric vertical – the same offset can be applied to a vertical CRS as to the height component of a three dimensional CRS.

Discussion: The horizontal displacement can be represented in units of metres or degrees in the east and north directions. Typically the underlying coordinate system will be a geographic coordinate system using latitude and longitude in which case it is computationally more efficient to apply a displacement defined in degrees. However most models are in terms of metres. Often this is because the models are derived from geophysical data which naturally are in metres. Metres are more intuitive to understand, particularly at high latitudes. Also degrees are unsuitable for defining uncertainty of horizontal deformation as latitude and longitude horizontal uncertainty are not directly comparable. Using metres allows displacements and corresponding uncertainties to be defined in the same units.

Discussion: The uncertainty can be represented in a different component to the displacements it relates to. There are two reasons this might be done. Firstly the resolution of grids required to define uncertainty may be much less than that required to define displacement as it is less

precisely defined. Secondly it means that applications that do not use the uncertainty information do not have to load grids of uncertainty values.

Deformation model attributes

The following attributes apply to the entire deformation model.

- Deformation model version
- Deformation model description
- Model role. Either point motion model or time dependent transformation.
- Source coordinate reference system. Defines the coordinate reference system (CRS) in terms of which the source coordinates are defined.
- Target coordinate reference system. For point motion models this must be the same as the source coordinate reference system.
- Spatial model coordinate reference system. Defines the coordinate system in which the spatial models (grid points) are defined. This proposal will require that this is the same as the source coordinate reference system.
- Reference epoch. Only applies to point motion models, defines the epoch at which the function evaluates to zero.
- Uncertainty reference epoch. This is the epoch relative to which uncertainties are calculated. This may be different to the model reference epoch. As an (theoretical) example, in New Zealand the deformation model includes a velocity with reference epoch 2000.0, so in principle error at epoch 2019 would be 19 times the uncertainty in the velocity (which is expressed in metres per year). However in practice the New Zealand geodetic control network was adjusted in 2018, when the order 0 (highest accuracy) control stations were accurately located by CORS observations, and the rest of the network was adjusted to bring it into alignment with these stations. The CORS stations NZGD2000 coordinates were calculated from the ITRF coordinates using the deformation model. So in effect the deformation model and geodetic control were recalibrated at 2018. So the error in 2019 due to the velocity component is only 1 times the annual uncertainty. In this case the uncertainty reference epoch is 2018.
- Horizontal deformation parameter specification. Defines how the deformation is parameterised. This proposal will support defining movement in north and east directions either in metres or degrees (degrees cannot be used for a projected source coordinate reference system).
- Vertical deformation parameter specification. This proposal support defining movement upwards in metres.
- Horizontal uncertainty parameter specification. Defining how horizontal displacement uncertainties are specified. This proposal will support circular 95% confidence limits in metres.
- Vertical uncertainty parameter specification. Defines how vertical uncertainties are specified. This proposal will support using 95% confidence limits in metres.
- Extents of deformation model. Defines the full extents for which the deformation model is defined. Beyond this extent the deformation is undefined. Component spatial models are assumed to zero within this extent if they do not explicitly define a deformation value. This proposal will support a bounding box in terms of the spatial model coordinate reference system.
- Grid interpolation method. This proposal will only support bilinear interpolation.
- Horizontal offset calculation method. This defines how offsets are calculated and added to horizontal coordinates. This proposal supports two methods, addition and geocentric. Additive uses simple addition of calculated values to the coordinates. For

geographic coordinate systems where the offsets are defined in metres this will use a local metre to degrees conversion for longitude and latitude. The geocentric method is an alternative method for geographic coordinate systems that works in high latitudes (where simple addition does not work). It is only applicable if the offsets are defined in metres. The geocentric method is described in detail below.

Component attributes

Each component is defined by a spatial model which evaluates to one or more of the deformation parameters at a location, and a time function to determine a scale factor by which it is multiplied.

The spatial model is defined by one or more grids. Grids may have different extents and different resolutions. At any given location a preferred grid will be selected and used to calculate the deformation. The grids should implement a nested structure conforming with the following rules.

- The grids are arranged in a tree structure defined by a single base grid and parent-child relationships
- Every grid is the either the base grid or the child grid of a single parent grid
- The spatial extents of a child grid lie within the spatial extents of their parent
- Child grids of a common parent do not overlap though they may share a common edge
- Where child grids meet at an edge they must evaluate to the same value along that edge
- The preferred grid at a location is the most deeply nested child including the location

The purpose of these rules is to ensure that the spatial function is unambiguously defined at any location.

The software should assume that the grids define a continuous deformation field. Where the model transitions from one grid to another the values calculated for deformation on each grid should be the same. Where the component does not cover the full extents of the deformation model it should evaluate to zero at the internal boundaries to ensure continuity.

Also the deformation field should be invertible within a reasonable range of times (velocity fields will not be invertible if projected too far). It is the responsibility of agencies creating deformation models to ensure these conditions are met - this could be supported by quality checking tools).

The handling of complexities in grid definition such as spanning the anti-meridian and definition in polar regions will be inherited from the base grid format and is not discussed here. However the extents of child grids should all lie numerically within the extents of their parent grids, and each grid should lie numerically within the extent of the parent.

The proposed interpolation method is to use bilinear interpolation. This will ensure that the displacement field is continuous across a grid, but it means that the derivatives of the displacement with respect to longitude are not continuous. The derivatives of horizontal deformation are strain values. These are mainly of interest to geophysicists. This model is not intended for geophysical use (though it may be derived from a geophysical model), so a discontinuous strain field is not seen as an issue.

The derivative of the vertical deformation with respect to longitude and latitude is tilt, which may be of interest to users (for example what is the impact of deformation on waste water systems) so a spatially discontinuous tilt field may be less desirable.

However for this proposal bilinear interpolation is preferred over more complex options such as bicubic interpolation which would generate a continuous tilt field across a grid. Note that even if bicubic interpolation was used it would not necessarily result in a continuous tilt field across the transition between the constituent grids of a component.

Each component has a defined spatial extent. If this does not include the calculation point then the spatial model evaluates to zero. Also if the component grids do not include the

calculation point it evaluates to zero. The component extent is provided to avoid the need to inspect all the grids.

Time function attributes

Each component of the model has a time function.

Time functions are either piecewise linear functions or exponential decay functions.

Piecewise linear functions are defined by an ordered set of time/date values and a corresponding set of scale factors defining the value by which the spatial model is multiplied at that time. The functions are not necessarily continuous – for example the model may define step function. The date/time values should be increasing. Where there is a step function the series will include two consecutive identical date/time values.

Four common simplifications of the general piecewise model are supported to simplify model definition. Each requires just a single reference date/time value. These are:

- constant displacement. The spatial model defines a constant offset. The reference date/time is ignored. This means that distortion grids and vertical offset grids could be encoded in this format as specialisations if that were desired.
- velocity model. The spatial model is assumed to define the displacement per year. The value is 0 at the reference date/time.
- Step function. The spatial model defines the value after the reference date/time. The value is zero before the reference date/time.
- Reverse step function. The spatial model defines the value before the reference date/time. The value is zero after the reference date/time.

The data required to define a piecewise time function for each component may therefore include:

- function type – one of constant, velocity, step, reverse step, piecewise linear.
- reference date/time – Used for velocity, step, and reverse step functions, ignored otherwise.
- date/time and scale factor pairs – ordered sequence of date/time and scale factors values. Only used by the piecewise linear function type
- behaviour before the first date/time value – one of zero, constant, linear. Only used by the piecewise linear function type
- behaviour after the last date/time value – one of zero, constant, linear. Only used by the piecewise linear function type

Exponential functions are defined by the following attributes:

- function type – exponential.
- reference date/time.
- end date/time - optional date to limit the time range over which the value changes
- relaxation constant measured in years.
- before scale factor - a constant value applying before the reference epoch. This allows for defining exponential “reverse patches”.
- initial scale factor - applies at the reference date/time
- final scale factor - the value which the time function asymptotically approaches.

Discussion: Geophysical deformation may be approximated by other functions, such as logarithmic functions and cyclic functions. They are also used in point deformation models, for example reference station coordinates in the International Terrestrial Reference Frame. This is commonly done in geophysical analysis. Currently these are not used in deformation models for coordinate systems. However including a piecewise linear function in this specification allows the options for modelling any displacement time behaviour to an arbitrary level of precision.

Deformation model format

The deformation model format is intended to provide an open publication format that can be directly and efficiently used by transformation software. This will at least require a binary grid format. The additional attributes and time function definitions are relatively small and either a text or binary format could be used directly by software.

The recommended format encodes deformation model is encoded in a set of files – one GeoTIFF nested grid file for each component and a JSON formatted master file holding the deformation model attributes, component attributes, and time function specifications. The master file specifies the name of each grid file. The software assumes that the grid file is retrieved from the same location (directory or parent url) as the master file.

Master file format

The high level JSON format will be three level hierarchical structure in which the top level holds the attributes of the entire model. This includes an array of components. Each component includes a spatial_model and a time_model attribute. These are detailed below.

The master file will reference the GeoTIFF grid files defining the spatial model for each component. These will be defined by file location relative to the master file, and an MD5 checksum that can be used to confirm that the correct grid file is being used. (The checksum is available to allow software validation of the model but is not expected to be used in PROJ or other evaluation software). More than one component can refer to the same grid file, though typically each component will use a unique grid file.

The master file includes sufficient information to determine whether a component is needed to be used to evaluate the model. The component GeoTIFF file only needs to be accessed if it is actually required. In particular the extents of the grid file will be specified in the master file.

The attributes defining the model are illustrated by the following example, which includes three deformation model components illustrating the different types of time models. (It does not include a “reverse_step” time function, which is very similar to the “step” type function, or an “exponential” function.).

```
{
  "file_type": "deformation_model_master_file",
  "format_version": "1.0",
  "name": "NZGD2000 deformation model",
  "version": "20180701",
  "publication_date": "2018-07-01T00:00:00Z",
  "license": "Create Commons Attribution 4.0 International"
  "description": "New Zealand Deformation Model.\nDefines the secular
...",
  "authority": {
    "name": "Land Information New Zealand",
    "url": "http://www.linz.govt.nz",
    "address": "Level 7, Radio New Zealand House\r\n155 The
Terrace\r\nPO ...",
    "email": "customersupport@linz.govt.nz"
```

```
  },
  "links": [
    {
      "href": "https://www.linz.govt.nz/nzgd2000",
      "rel": "about",
      "type": "text/html",
      "title": "About the NZGD2000 deformation model"
    },
    {
      "href":
"https://www.geodesy.linz.govt.nz/download/nzgd2000_deformation_mode
...",
      "rel": "source",
      "type": "application/zip",
      "title": "Authoritative source of the NZGD2000 deformation mode
..."
    },
   {
     "href": "https://creativecommons.org/licenses/by/4.0/",
     "rel": "license",
     "type": "text/html",
     "title": "Creative Commons Attribution 4.0 International license"
   },
    {
      "href":
"https://www.geodesy.linz.govt.nz/download/nzgd2000/metadata/nzgd2000
...",
      "rel": "metadata",
      "type": "application/xml",
      "title": " ISO 19115 XML encoded metadata regarding the
deformation ..."
    }
  ],
  "source_crs": "EPSG:4959",
  "target_crs": "EPSG:7907",
  "definition_crs": "EPSG:4959",
  "reference_epoch": "2000-01-01T00:00:00Z",
  "uncertainty_reference_epoch": "2018-12-15T00:00:00Z",
  "horizontal_offset_unit": "metre",
  "vertical_offset_unit": "metre",
  "horizontal_uncertainty_type": "circular 95% confidence limit",
  "horizontal_uncertainty_unit": "metre",
  "vertical_uncertainty_type": "95% confidence limit",
  "vertical_uncertainty_unit": "metre",
  "horizontal_offset_method": "addition",
  "extent": {
    "type": "bbox",
    "parameters": {
        "bbox": [158.0,-58.0,194.0,-25.0]
      }
  },
  "time_extent": {
    "first": "1900-01-01T00:00:00Z",
    "last": "2050-01-01T00:00:00Z"
  },
  "components": [
```

```
    {
      "description": "Secular deformation model derived from NUVEL-1A
rotation ...",
      "displacement_type": "horizontal",
      "uncertainty_type": "none",
      "horizontal_uncertainty": 0.01,
      "vertical_uncertainty": 0.01,
      "extent": {
        "type": "bbox",
        "parameters": {
            "bbox": [158.0,-58.0,194.0,-25.0]
          }
      },
      "spatial_model": {
        "type": "GeoTIFF",
        "interpolation_method": "bilinear",
        "filename": "nzgd2000-ndm-grid02.tif",
        "md5_checksum": "49fce8ab267be2c8d00d43683060a032"
      },
      "time_function": {
        "type": "velocity",
        "parameters": {
          "reference_epoch": "2000-01-01T00:00:00Z"
        }
      }
    },
    {
      "description": "Event: Kaikoura earthquake,  14 November 2016\n
Source ...",
      "displacement_type": "horizontal",
      "uncertainty_type": "none",
      "horizontal_uncertainty": 0.01,
      "vertical_uncertainty": 0.01,
      "extent": {
        "type": "bbox",
        "parameters": {
            "bbox": [165.85,-47.625,179.05,-34.0]
          }
      },
      "spatial_model": {
        "type": "GeoTIFF",
        "interpolation_method": "bilinear",
        "filename": "nzgd2000-ka20161114-grid01.tif",
        "md5_checksum": "2f801a2ca3a46ae1d08c69bec2d47b5d"
      },
      "time_function": {
        "type": "step",
        "parameters": {
          "step_epoch": "2016-11-14T00:00:00Z"
        }
      }
    },
    {
      "description": "Event: Kaikoura earthquake postearthquake month
0-1, ...",
      "displacement_type": "horizontal",
```

```
      "uncertainty_type": "none",
      "horizontal_uncertainty": 0.01,
      "vertical_uncertainty": 0.01,
      "extent": {
        "type": "bbox",
        "parameters": {
            "bbox": [169.6,-43.75,176.35,-39.0]
          }
      },
      "spatial_model": {
        "type": "GeoTIFF",
        "interpolation_method": "bilinear",
        "filename": "nzgd2000-ka20161114-grid04.tif",
        "md5_checksum": "97c52773684695f6126b4b34ded92e85"
      },
      "time_function": {
        "type": "piecewise",
        "parameters": {
          "before_first": "zero",
          "after_last": "constant",
          "model": [
            {"epoch":"2016-11-14T00:00:00Z","scale_factor":0.0},
            {"epoch":"2016-12-14T00:00:00Z","scale_factor":1.0}
          ]
        }
      }
    }
  ]
}
```

The root element of the JSON file contains the following elements (elements not required to calculate the model are *italicised*. The are required in the format to support future extensibility and provide metadata identifying the authority of the model, etc):

- file_type. Always “deformation_model_master_file”
- format_version. Specifies the version of the format. This proposal specifies the 1.0 format. Future versions could add other features, such as triangulated spatial models, other time function types, and so on.
- *name*. A brief descriptive name of the deformation model.
- *version*. A string identifying the version of the deformation model. The format for specifying version will be defined by the agency responsible for the deformation model.
- *publication_date*. The date on which this version of the deformation model was published (or possibly the date on which it takes effect?)
- *description*. A text description of the model.
- *authority*. Basic information about the agency responsible for the data set. Includes elements:
    - name. The name of the agency
    - url. The url of the agency website
    - address. The postal address of the agency
    - email. An email contact address for the agency
- *links*. (optional) Links to related information. Each link contains the following elements:
    - href. The URL holding the information
    - rel. The relationship to the dataset. Proposed relationships are:
        - “about”, a web page for human consumption describing the model
        - “source”, the authoritative source data from which the deformation model is built
        - license, the licence under which the model is published

            - "metadata". ISO 19115 XML metadata regarding the deformation model.
        - type. The content type of the resource
        - title. A title describing the linked content
- source_crs. The coordinate reference system to which the deformation model applies
- target_crs. For a time dependent coordinate transformation the coordinate reference system resulting from applying the deformation. It is assumed that the target system is identical to the source system except that its underlying datum differs by the modelled deformation. If they are projected coordinate systems then they will use the same projection formulae and parameters.
- definition_crs. The coordinate reference system used to define the component spatial models. Note that the offset computed defined by the spatial model is in terms of the source_crs. For example a grid could be defined in a local projection, but it could determine offsets to apply to a geographic coordinate system (for which the direction of east and north might be different). For the initial version of the format it is required that the definition system is the same as the source coordinate system.
- *reference_epoch*. A nominal reference epoch of the deformation model. This is not necessarily used to calculate the deformation model - each component defines its own time function.
- uncertainty_reference_epoch. The uncertainties of the deformation model are calculated in terms of this epoch. This is described below in the "Time functions" section.
- *horizontal_offset_unit*. This proposal will only support "metre" and "degree".
- *vertical_offset_unit*. This proposal will only support "metre".
- *horizontal_uncertainty_type*. This proposal will only support "circular 95% confidence limit",
- *horizontal_uncertainty_unit*. This proposal will only support "metre",
- *vertical_uncertainty_type*. This proposal will only support "95% confidence limit",
- *vertical_uncertainty_unit*. This proposal will only support "metre",
- horizontal_offset_method. Defines how the horizontal offsets are applied to geographic coordinates. This proposal supports two methods, "addition" and "geocentric",
- extent. Defines the region within which the deformation model is defined. It cannot be calculated outside this region. The region is specified by a type and value. This proposal only supports using a bounding box as an array of [west,south,east,north] coordinate values.
- time_extent. Defines the range of times for which the model is valid, specified by a first and a last value. The deformation model is undefined for dates outside this range.
- components. An array of the components comprising the deformation model.

Each component contains the following elements:

- *description*. A text description of this component of the model.
- extent. The region within the component is defined. Outside this region the component evaluates to 0. The region is specified by a type and value. This proposal only supports using a bounding box as an array of [west,south,east,north] coordinate values
- displacement_type. The displacement parameters defined by the model, one of "none", "horizontal", "vertical", and "3d". The "none" option allows for a component which defines uncertainty with different grids to those defining displacement.
- uncertainty_type. The uncertainty parameters defined by the model, one of "none", "horizontal", "vertical", "3d"
- horizontal_uncertainty. The horizontal uncertainty to use if it is not defined explicitly in the spatial model.
- vertical_uncertainty. The vertical uncertainty to use if it is not defined explicitly in the spatial model.
- spatial_model. Defines the spatial model with the following elements:
    - type. Specifies the type of the spatial model data file. Initially it is proposed that only GeoTIFF is supported.
    - interpolation_method. This proposal will support "bilinear" and

"geocentric_bilinear",
    - filename. Specifies location of the spatial model GeoTIFF file relative to this JSON file.
    - *md5_checksum*. A hex encoded MD5 checksum of the grid file that can be used to validate that it is the correct version of the file.
- time_function. Defines the time function of the component. This has the following elements:
    - type: one of "constant", "velocity", "step", "reverse_step", "piecewise", "exponential"
    - parameters: the parameters used to quantify the time function.

The five types of time function are illustrated in the example JSON above, apart from exponential. The parameters of each are:

- constant. This has no parameters - its value is always 1.0.
- velocity. This has just one parameter, "reference_epoch" which is the date/time at which the velocity function is zero.
- step. This has one parameter, "step_epoch", at which the step function transitions from 0 to 1.
- reverse_step. This has one parameter, "step_epoch", at which the reverse step function transitions from -1 to 0.
- piecewise. The piecewise function has three parameters:
    - before_first. One of "zero", "constant", and "linear", defines the behaviour of the function before the first defined epoch.
    - after_last. One of "zero", "constant", and "linear", defines the behaviour of the function after the last defined epoch.
    - model. A sorted array data points each defined by two elements, "epoch" defines the date/time of the data point, and "scale_factor" is the corresponding function value. The array is sorted in order of increasing epoch. Note: where the time function includes a step it is represented by two consecutive data points with the same epoch. The first defines the scale factor that applies before the epoch and the second the scale factor that applies after the epoch.
- exponential. The exponential function has the following parameters:
    - reference_epoch. The date/time at which the exponential decay starts
    - end_epoch (optional). The date/time at which the exponential decay ends
    - relaxation_constant. The relaxation constant in years
    - before_scale_factor. The scale factor that applies before the reference epoch
    - initial_scale_factor. The initial scale factor
    - final_scale_factor. The scale factor the exponential function approaches. Note that if an end_epoch is specified this value will never be reached.

Discussion: Format of components

For several of the elements of the model there is a choice of how the values should be represented in JSON.

Date/time values are formatted as *YYYY-MM-DD*T*hh:mm:ss*Z. This is directly readable by many applications and is ISO 8601 compliant. In practice it is rare that the time component is meaningful or required - a simple date would suffice. This format specifies the time as UTC. To simplify implementation it is proposed that all dates/times are nominally UTC.

Internally the format used by PROJ for times is decimal years. The reasons for choosing a calendar date format are that 1) this format is more generally used, and it is hoped that this format could be used by other software stacks, and 2) using calendar dates is more easily human readable. The main place dates will be used in the format is for episodic events. These are usually relating to earthquakes which are commonly identified by calendar dates. Using this format makes it easier to visually inspect the file for correctness.

The calculation notes below define how calendar dates are to be converted to decimal years and the implications of doing so.

The model_crs is defined by authority:id. EPSG:id is used in this example and is probably the most suitable convention for use in the near future. It is recommended that for the moment EPSG codes are used. Note that a deformation model could be used for both a 2 dimensional

geographic CRS, a 3 dimensional CRS, or a 2 dimensional geographic CRS + vertical CRS. Where these share a common datum it is proposed that they are used interchangeably if that is possible.

For example in this case the model_crs is EPSG:4167, a 2 dimensional CRS. However this should be usable with the corresponding 3 dimensional CRS EPSG:4959.

GeoTIFF parameters

The GeoTIFF format for the spatial model grid files is described in PROJ RFC4 (https://github.com/rouault/PROJ/blob/rfc4_remote_and_geotiff_grid/docs/source/community/rfc/rfc-4.rst#description-of-the-proj-geotiff-format).

To encode the deformation model requires a minor amendment to the GeoTIFF specification to include deformation format grid types and sample types in the metdata.

It is proposed that the deformation model format is a single GeoTIFF file type. However it encapsulates 15 alternative subtypes according to which deformation parameters are included. The displacement and uncertainty parameters can each be one of "none", "horizontal", "vertical", or "3d". The combination "none", "none" is not valid as it has no sample values at grid node. Some other combinations are nonsensical, for example "horizontal" displacements with "vertical" uncertainties.

It is proposed that these are encoded into GDAL metadata as

```
<Item name="TYPE">DEFORMATION_MODEL</Item>
<Item name="DISPLACEMENT_TYPE">HORIZONTAL</Item>
<Item name="UNCERTAINTY_TYPE">NONE</Item>
```

The other extension to the GeoTIFF format is the sample types defined at each node. This deformation model will support the following sample types:

- east_offset. Required if DISPLACMENT_TYPE is HORIZONTAL or 3D. Units metre or degrees.
- north_offset. Required if DISPLACMENT_TYPE is HORIZONTAL or 3D. Units must be the same as used for east offset.
- vertical_offset. Required if DISPLACEMENT_TYPE is VERTICAL or 3D. Units must be metre.
- horizontal_uncertainty. Required if UNCERTAINTY_TYPE is HORIZONTAL or 3D. Units must be metres.
- vertical_uncertainty. Required if UNCERTAINTY_TYPE is VERTICAL or 3D. Units must be metres.

Following the GeoTIFF proposal these are encoded into the GDAL metadata as (for example):

```
<Item name="DESCRIPTION" sample="0" role="description">east_offset</Item>
<Item name="UNITTYPE" sample="0" role="unittype">metre</Item>
<Item name="DESCRIPTION" sample="1" role="description">north_offset</Item>
<Item name="UNITTYPE" sample="1" role="unittype">metre</Item>
<Item name="DESCRIPTION" sample="2" role="description">vertical_offset</Item>
<Item name="UNITTYPE" sample="2" role="unittype">metre</Item>
```

Each sample type will also have a SCALE and OFFSET value as specified in the GeoTIFF grid format.

Note that some of the grid parameters are duplicated in the JSON master file. Applications using the model can check these are consistent and fail if they are not. Consistency is the responsibility of the agency publishing the deformation model.

All sample values will evaluate to zero beyond the range of the grid when it is used as part of a deformation model. Evaluating beyond the range of the grid(s) will not cause a failure.

Utilities will be built to check consistency between master file and constituent grid files.

Alternative implementations

The initial intention of this proposal was to define a format which would encode the entire deformation in a single binary file. This would be similar to the idea raised as a PROJ enhancement request in 2018 ((https://github.com/OSGeo/PROJ/issues/1001). The GeoTIFF format could be extended using the TIFF GDAL_METADATA tag (https://www.awaresystems.be/imaging/tiff/tifftags/gdal_metadata.html) to encode the attributes of the deformation model and for individual components. The main features of this approach are:

- the GDAL metadata entry for each grid would include a component id entry to identify which component the grid belongs to
- the deformation model attributes would be encoded in the GDAL metadata of the first grid in the file.
- the component attributes (including time function) would be encoded in the GDAL metadata of the first grid of the component.

The advantage of the single file format is that it is easy to distribute without a risk of implementing an incomplete or otherwise corrupt model. It is also easy to confirm if the model is available by checking for the presence of this file.

However the multiple file format offers many advantages. These include:

- the component GeoTIFF files can be interpreted by the code written for nested grid distortion and vertical shift files very little or no modification. Similarly the tools built to support these formats would require little modification.

- agencies interested in a subset of the area covered by the model may not need to acquire the entire model. For example an agency based in the eastern US need not ever obtain the files defining the deformation on the west coast.

- the JSON format for the model and component attributes is very easy to generate and view

- it is a much cleaner implementation. Shoe-horning the metadata into a GeoTIFF format would result in a messy implementation that only specialised tools could handle.

- grid files could be used in multiple models. For example after an earthquake there may be a new version of deformation model that adds a component to an existing model. Most components of the deformation model would be common to the old and new versions. Sharing grid files would avoid duplication of data. This could also allow optimising when converting between versions of the datum, as components common to both versions could be ignored (indeed this is necessary to properly calculate the uncertainty of the transformation).

- it is readily extensible to use other grid formats, or even spatial models which are not gridded.

The risk of obtaining a corrupt model (e.g. wrong version of a grid file) is mitigated by including a checksums for each grid file in the master file. These can be used for model validation.

The model can still be published as a single file format, for example using a zip archive format in the same way as proj-datumgrid resources currently are. However the model would need to be extracted to be used.

The format of the master file is as a JSON file. A binary encoded format would be faster to load. However this is not a large data set. For example the current New Zealand model with 24 components is about 24kb in size. So parsing would be very quick - certainly compared to the computations of evaluating the model at more than a few points.

The JSON schema proposed could be simpler than it is. Many attributes are included which are not required to calculate the model or which have only one permitted value. These are included to make the file more readable and self describing, and to ensure the format is

extendable to support future development of deformation models without invalidating these proposed format. Again the overhead from the extra attributes is considered insignificant.

Other candidate formats considered were XML and CSV. The XML format was discarded as being unnecessarily verbose (slightly more time to load), and harder to read. CSV was considered as being a very simple format to parse. However it does not suit the hierarchical structure of the model.

From the view of the PROJ software JSON is a good option as it already supports parsing JSON formatted data. It is also very amenable to using in web services.

Calculation formulae

Grid interpolation

Bilinear interpolation

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

$x_i = x_o + i.x_s$

$y_j = y_o + j.y_s$

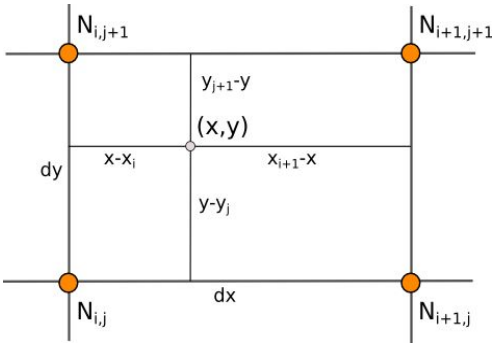
where x_o, y_o are the longitude and latitude of the southwest-most corner of the grid, x_s and y_s are the longitude and latitude grid spacing, and i and j are the column and row number of the grid cell (where the west-most column and southernmost row are numbered 0). Note that the longitude and latitude grid spacing need not be equal – it is preferred that x_s is approximately equal to $y_s/\cos(y_m)$, where y_m 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:

$W_{i,j} = ((x_{i+1}-x)/x_s) * ((y_{j+1}-y)/y_s)$

$W_{i+1,j} = ((x-x_i)/x_s) * ((y_{j+1}-y)/y_s)$

$W_{i,j+1} = ((x_{i+1}-x)/x_s) * ((y-y_j)/y_s)$

$W_{i+1,j+1} = ((x-x_i)/x_s)*((y-y_j)/y_s)$

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

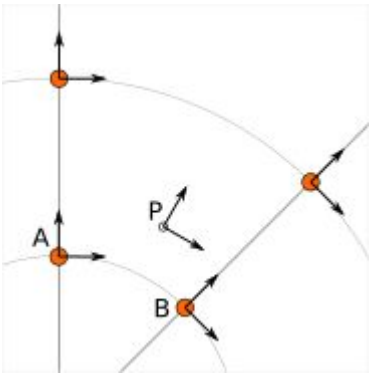
$de = W_{i,j}*de_{i,j} + W_{i+1,j}*de_{i+1,j} + W_{i,j+1}*de_{i,j+1} + W_{i+1,j+1}*de_{i+1,j+1}$

The error elements eh, ev are interpolated using a weighted average of the variances eh^2, ev^2, for example

$eh = \sqrt{(W_{i,j}*eh_{i,j}^2 + W_{i+1,j}*eh_{i+1,j}^2 + W_{i,j+1}*eh_{i,j+1}^2 + W_{i+1,j+1}*eh_{i+1,j+1}^2)}$

“Geocentric bilinear” interpolation

A model using a latitude/longitude grid may not be appropriate for models including polar regions. The simplest approach to defining a model in a polar region may be to use a suitable projected coordinate system for the definition.



However a latitude and longitude grid can still serve if the north and east component of deformation are defined in metres. The simple bilinear interpolation method above is less reasonable if the east and north vectors at each grid node are not in approximately the same direction. As shown in the figure this may not be the case near the pole if the longitude grid spacing is large.

An interpolation alternative approach that can be used is to convert the displacement components from east and north components to geocentric X, Y, and Z components. These are in the same direction can be scaled and summed using the bilinear formulae above to calculate the X, Y, Z components of displacement at the calculation point, which are then converted back to components east and north at the calculation point.

Note that this is only used to determine the horizontal displacement. The vertical displacement can still be computed using the formulae above

At longitude λ and latitude ϕ the dx, dy, dz values are calculated from the east and north displacements de, dn as:

$$dx_{i,j} = -de_{i,j}.\sin(\lambda_{i,j}) - dn_{i,j}.\cos(\lambda_{i,j}).\sin(\phi_{i,j})$$
$$dy_{i,j} = de_{i,j}.\cos(\lambda_{i,j}) - dn_{i,j}.\sin(\lambda_{i,j}).\sin(\phi_{i,j})$$
$$dz_{i,j} = dn_{i,j}.\cos(\phi_{i,j})$$

The X, Y, and Z directions are the same at any location, so the dx, dy, and dz displacements can be interpolated independently using bilinear interpolation as described above, eg:

$$dx = W_{i,j}*dx_{i,j} + W_{i+1,j}*dx_{i+1,j} + W_{i,j+1}*dx_{i,j+1} + W_{i+1,j+1}*dx_{i+1,j+1}$$

The displacement at the calculation point is then calculated as:

$$de = -dx.\sin(\lambda) + dy * \cos(\lambda)$$
$$dn = -dx.\cos(\lambda).\sin(\phi) - dy.\sin(\lambda).\sin(\phi) + dz.\cos(\phi)$$

Discussion: To estimate the error that could be incurred by not accounting for this difference in direction we can consider a case where the deformation is 1m northwards at A, and zero at B. Let the longitude grid spacing be λ_s radians. If the calculation point P is λ radians past A, then the magnitude of the interpolated vector will be $(\lambda_s - \lambda)/\lambda_s$. The error of orientation will be λ radians (the difference between north at A and north at the calculation point). So the vector error will be $\sin(\lambda).(\lambda_s - \lambda)/\lambda_s$. Approximating $\sin(\lambda)$ as λ, this has a maximum absolute value in the range $(0, \lambda_s)$ of $\lambda_s/2$. So for example with a grid longitude spacing of 1° this could result in a 2cm error in the 1m of deformation vector.

Using the geocentric interpolation method to calculate the horizontal component does cause some “leakage” of the horizontal deformation into the vertical component, that is:

$$du = dx.\cos(\lambda).\cos(\phi) + dy.\sin(\lambda).\cos(\phi) + dz.\sin(\phi)$$

For the interpolation of du this method is using the same formulae as the bilinear interpolation method, that is simple bilinear interpolation of the du component. However this leakage does result in a small loss of magnitude in the horizontal component. The reduction is approximately scaling by the cosine of the angle between the vertical at the calculation point and the vertical at each grid node. For a grid cell of 1 degree extent this would result in a scale error of 0.2mm for a 1m deformation vector, so this is ignored. (Note that this is a 1 degree extent measured on the globe - not a 1 degree extent of longitude which may be much smaller near the poles).

Time functions

The time function for a component defines a scale factor f(*t*) applied to component displacements at time *t*.

Following conventional use in deformation models all date/time values are converted to decimal years for use in the following formulae. The conversion to decimal years is done by first converting all dates to UTC. The year number *yyyy* of the epoch forms the integer part of the decimal year. The fractional part of the epoch is determined by dividing the number of seconds between *yyyy*-01-01T00:00:00Z and the epoch by the number of seconds between *yyy1*-01-01T00:00:00Z and *yyyy*-01-01T00:00:00Z, where *yyy1* is *yyyy*+1.

For the uncertainties eh, ev the scale factor $f_e(t)$ is defined as $\sqrt{\text{abs}(f(t)-f(t_0))}$ where t_0 is the uncertainty reference epoch of the model.

For the constant function type $f(t) = 1$.

For function types velocity, step, and reverse step the model is defined by a reference time t_0. For these types f(*t*) is defined as:

velocity	$f(t) = (t - t_0)$ all values of t
step	$f(t) = 0$ when $t < t_0$,
	$f(t) = 1$ when $t >= t_0$
reverse step	$f(t) = -1$ when $t < t_0$,
	$f(t) = 0$ when $t >= t_0$

For function type "piecewise" the function is defined by a series of times t_i and multiplying factors f_i for i=1 to n. The time function is defined as:

piecewise	$f(t) = (f_i.(t_{i+1} - t) + f_{i+1}.(t-t_i))/(t_{i+1}-t_i)$
	where $t_i <= t < t_{i+1}$

The value before t_1 and after t_n depends on the behaviour specified and is defined as follows:

behaviour	Before t_1	After t_n
zero	0	0
constant	f_1	f_n
linear	$f(t) = (f_1.(t_2 - t) + f_2.(t-t_1))/(t_2-t_1)$	$f(t) = (f_{n-1}.(t_n - t) + f_n.(t-t_{n-1}))/(t_n-t_{n-1})$

For function type "exponential" the function is defined by a reference epoch t_0, an optional end epoch t_1, a relaxation constant θ, and three scale factors, before f_p, initial f_0, and final f_∞. The scale factor at time *t* is defined as:

$f(t) = f_p$ when $t < t_0$

$f(t) = f_0 + (f_\infty - f_0).(1 - \exp(-(t-t_0)/\theta))$ when $t_0 <= t < t_1$

$f(t) = f_0 + (f_\infty - f_0).(1 - \exp(-(t_1-t_0)/\theta))$ when $t >= t_1$

Combination of components

To calculate the total deformation at a time and location, the displacement and uncertainties due to each component are calculated independently and then summed together to obtain the total displacement at a location. This displacement is then applied to the coordinate.

The same input coordinate is used for each component - the components are not applied sequentially (ie the coordinate is not updated by the first component before being used to calculate the deformation on the second component).

At a given time and location the elements from each component 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 de_1, de_2, ... to de_n, and the time function evaluates to f_1, f_2, ... to f_n then the total model value for de is

$$de = f_1.de_1 + f_2.de_2 + \ldots + f_n.de_n$$

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

$$eh = \sqrt{(f_{e,1}^2.eh_1^2 + f_{e,2}^2.eh_2^2 + \ldots + f_{e,n}^2.eh_n^2)}$$

One extra subtlety that may occur in calculating errors is that more than one component may use the same grid file. In this case the scale factors for the components using the grid are simply added together before being combined with the other components using RSS.

Discussion. An alternative approach that could be used is to apply components sequentially. That is the first component is calculated and applied to the coordinate, and then the modified coordinate is used to calculate the second component, and so on. This may result in a different final coordinate to the proposed method, as the second and subsequent components are evaluated at a different location.

Neither method is more correct from a theoretical point of view. The main reason for specifying one approach is to ensure that there is an "authoritative" correct value, particularly where the deformation model is used in the definition of a datum (as in New Zealand for example).

If the components are an ordered sequence of discrete events then the sequential approach might seem more intuitive. However this is not necessarily the case. For example consider a model in which the first component is a velocity function and the second is a step at 2003-01-01. If the deformation is calculated at 2004-01-01, the velocity function is applied as at 2004, and then that coordinate is used for the step function. If the deformation is calculated at 2014-01-01, then the velocity function is applied as at 2014, and that different coordinate is used to interpolate the step function model. This means that the contribution from the step function could be different even though nothing else has changed other than the evaluation epoch.

In practice the choice of independent or sequential evaluation of components is very unlikely to make a significant difference to the coordinates - at worst it is very similar to that described below for the inverse method in relation to iterating the inverse calculation or not. The choice of independent evaluation has some small advantages in calculation in that:

- using the same input coordinates is slightly more efficient as the calculated displacement only needs to be applied to the coordinate once. This could be a significant difference if the horizontal displacement is applied using the "geocentric" method as described below. It is insignificant if the displacement is applied by simple addition.
- using the same input coordinates for all components provides an opportunity for parallelising calculation of components.
- using the same input coordinates for each component allows optimising transformations between two versions of the deformation model as common components can be ignored.

Calculation of the inverse deformation model

Calculating the inverse of the deformation model requires an iterative solution in that the coordinate in the source coordinate reference system is required to evaluate the deformation model, but it is not known until the deformation has been calculated and applied to the input coordinate in the target coordinate reference system.

The iteration is done by

- using the input coordinate as an initial estimate for the output coordinate

- at each iteration:
    - apply the deformation model to the current estimate of the output coordinate
    - calculate the difference between the calculated coordinate and the input coordinate
    - subtract this difference from the current estimate solution to obtain the estimated solution for the next iteration
    - if this difference is less than the precision required for the inverse operation then finish

The calculation of the difference and the subtraction of the difference from the current estimate is done by the "addition" or "geocentric" method, as defined in the deformation model header. (Formulae are defined below under "Applying the offset to a coordinate" and "Calculating horizontal deformation near the poles".

The error of not iterating the inverse transformation can be tested for the New Zealand NZGD2000 deformation model. The least smooth area of deformation in New Zealand is that affected by the 2016 Kaikoura earthquake. As this has been updated by "reverse patching" the inhomogeneity of the deformation field primarily affects pre-earthquake transformations. Testing across the fault zone finds that the maximum error from not iterating an inverse transformation of epoch 2000.0 coordinates is about 0.015 metres. However this is in an area where the deformation model is very inaccurate - it is smoothed across the fault zone and will have errors of many decimetres. For transforming epoch 2019.0 coordinates the maximum error is about 0.000014 metres. In the North Island in an area largely unaffected by episodic events the maximum error is about 0.00015 metres.

Based on this result it is recommended that the inverse transformation is iterated. It is likely that this will double computation time (it would be unusual to require more than two iterations).

Note that this is not about creating a more accurate transformation - the differences are much less than the uncertainty in the deformation model. The reason for iterating is to satisfy a user expectation that applying a transformation followed by the inverse transformation will result in coordinates that are materially unchanged.

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 t_0 and t_1 is calculated for each component separately as $f_{e,t1-t0} = \sqrt{}\text{abs}(f(t_1)-f(t_0))$.

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.

Applying the offset to a coordinate

Two methods are proposed for applying the calculate offset at a point to the coordinate, *addition* and *geocentric*. The *addition* method is relatively simple and simply adds the offset to the coordinates, converting metres to degrees first if necessary. The *geocentric* method is an alternative method that may be used near the poles if the grid latitude spacing is relatively large. It is only applicable if the offsets are defined in metres and the coordinate system is a geographic (latitude/longitude) system. In this case the horizontal offset is converted to a geocentric (XYZ) offset, added to the geocentric coordinate, and then converted back to

geographic coordinates. Note that the vertical coordinate is calculated in the same way as in the *additive* method. These methods are detailed below.

Addition method

The method of the calculated east/north/up displacement to a coordinate depends on the units of the displacement and the type of the source and target coordinate system. Also for geographic coordinate systems the method described here does not apply very close to the poles. See the section below "calculation horizontal deformation near the poles" for details.

If the source and target coordinate systems are projected coordinate systems then the units must be metres and the east and north displacements are simply added to the easting, northing ordinate.

If the source and target coordinate systems are geographic coordinate systems and the east and north displacement units are degrees, then again the displacements are added to the longitude and latitude.

If the source and target coordinate systems are geographic coordinate systems and the east and north displacement units are metres then the displacement components must be converted to degrees before they are added to longitude and latitude. The conversion from metres to degrees requires the ellipsoid parameters of the geographic coordinate system.

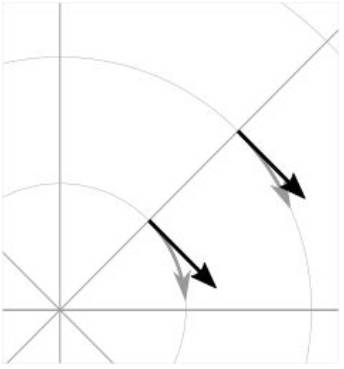
If a is the ellipsoid semi-major axis (eg 6378137.0), f is the flattening (eg 1.0/298.25722210), λ is the latitude, and ϕ is the longitude then corrections to longitude and latitude (in radians) are given by:

$$b = a.(1-f)$$
$$d\lambda = de.\sqrt{(b^2\sin^2(\phi)+a^2\cos^2(\phi))}/a^2\cos(\phi)$$
$$d\phi = dn.(b^2\sin^2(\phi)+a^2\cos^2(\phi))^{3/2}/a^2b^2$$

The vertical offset is always in metres and is simply added to the height coordinate.

Geocentric method

The geocentric method can be applied if the model is using a geographic coordinate system and offsets measured in metres..



This method may be applicable for coordinates near the pole, where simple addition of offsets to the longitude/latitude may not give the desired result. This is shown in the figure where the grey vector shows adding an offset to the longitude, and the black vector shows applying the offset as a vector offset to the coordinate. Close to the pole the eastward vector is different to changing the longitude coordinate.. In this case the maximum potential error from the approximate method is the size of the east displacement.

Moving away from the pole this issue rapidly becomes insignificant. For a point at distance R from the pole with a displacement d, the difference is approximately d*(1-cos(d/R)), or approximately $d^3/2R^2$. So for example a 1m eastward displacement 10km from the pole would have an error of 10^{-8}m. This is only an issue very close to the pole!

The geocentric method is an alternative method for applying the horizontal offset to overcome this issue. The horizontal offset and original longitude and latitude are both converted to a geocentric coordinate system. The offset is then added to the coordinate, and the result coordinate converted back to the geographic coordinate system. This defines the new latitude and longitude. The new height coordinate is calculated by adding the height offset to the original height coordinate in the same way as for the additive method.

Note that to be consistent with the additive method the calculation of the geocentric coordinate from the geographic coordinate ignores the height coordinate. It is replaced with an elevation of zero for this calculation.

Standard formulae are used for the conversion of geographic coordinates to and from geocentric coordinates. The initial ellipsoidal height is set to zero before converting, and the resultant ellipsoidal height is discarded.

The horizontal components of displacement are converted to X,Y,Z components using the same formulae as described for the "geocentric_bilinear" method.

While this method can be used at any location it is not recommended other than close to the poles. It is computationally very expensive compared to simply adding the offsets to longitude and latitude (particularly if the offset units are degrees). In particular the conversion from geocentric to geographic coordinates does not have a closed formula, so this calculation must be iterated to obtain the required accuracy for the conversion.

Implementation in PROJ

The deformation format is implemented in PROJ as a transformation step that can be invoked in a transformation pipeline with a step such as

```
+step +proj=defmodel +modelfile=nzgd2000_deformation_20180701.json
+step +inv +proj=defmodel +modelfile=nzgd2000_deformation_20180701.json
```

The master file would be located in the same way as grid files, including using the grid_alternatives table where the local name might differ from that defined in a coordinate repository such as EPSG.

The PROJ implementation encodes the handling of the master file and most of the calculation functions in a single C++ header file. Functions to retrieve coordinate system information and to evaluate GeoTIFF grids are interfaced to existing PROJ functionality. This approach means that the header file can be used without modification in other implementations. Currently this capability does not include calculating deformation uncertainty (as it is not used anywhere in PROJ).

The function may raise an error if the assumptions of the model are not met, for example if a grid does not provide the displacement components such as de, dn, dh that are defined for it in the master file.

Discussion. An earlier proposal considered the option of expanding the deformation model to use a PROJ pipeline using already existing options such as +proj=hgridshift and +proj=vgridshift for each component of the model.

This would have required some additional options to allow selection logic in the pipeline step, for example to include piecewise continuous time functions. Ideally it would also have needed conditional operators to handle skipping a calculation step if the calculation point is outside a specified coordinate range, so that transformation grids would not be loaded if they were not needed. The grid evaluation function would also need an option to allow the step to be skipped if the evaluation point lay outside the grid without generating an error.

The rationale for the preferred approach of building a specific deformation function is:

- this proposal prefers using independent calculation of the deformation of each component deformation rather than sequential calculation - the same input coordinate is used for each component (which would not be possible with the pipeline framework)
- a single deformation model base function could be useful for other applications. It could be expanded to support calculating deformation model uncertainty which other applications might require

Conversion of coordinates between versions of the deformation model

A common source of confusion is coordinate transformations between different versions of a datum.

For example in New Zealand the deformation model was recently updated from version 20171201 to 20180701. Technically this is equivalent to a new version of the datum.

Users with a GIS datasetin terms of the 20171201 version of the datum might want to update the dataset to version 20180701. The user expectation is that this will generate correct version 20180701 coordinates of the features in the database.

The critical thing in this transformation is that the coordinate epoch for the transformation is before the event(s) implemented in the update. This is somewhat counter-intuitive.

Generally the update should not change the coordinates. The reason for the update is typically a deformation event such as an earthquake. The earthquake coseismic deformation is added to the deformation model as a step function that applies for transforming coordinates for epochs after the event. This means that the NZGD2000 coordinate system tracks the movement of features fixed to the ground and therefore the NZGD2000 coordinates of these features are not changed by the earthquake. In this case the deformation model is unchanged before the earthquake. Transforming at an epoch before the earthquake will leave the coordinates unchanged which is what is required..

Close to faulting the distortion due to the earthquake can be too intense to be included in the coordinates. In that case the deformation model will be smoothed across the fault zone. However the deformation is still measured and is used to update the coordinates. It is also added to the deformation model using a reverse step function that applies a negative deformation that applies when transforming coordinates for epochs before the earthquake. In this case transforming coordinates at an epoch before the earthquake will result in subtracting the reverse patch from the coordinates. This adds the deformation to the coordinates, which again is the correct update to coordinates to transform them to the new version of the datum.

Outstanding issues

The use of the uncertainty reference epoch presents a difficulty from the point of view of maintaining the deformation model. The appropriate reference epoch for the uncertainty could change far more frequently than any other attribute of the model. For example in New Zealand the national geodetic is periodically recalculated using the most current ITRF coordinates of the reference stations. This will change the uncertainty reference epoch for the deformation model, but otherwise leave it unchanged. It is debatable whether this should constitute a new version of the deformation model, or of the datum it relates to. Since most users will not ever calculate or use the uncertainties it makes no practical difference.

Perhaps the most sensible approach for software that used the uncertainty information is that it should be able to override the uncertainty reference epoch.

Another alternative is to remove the uncertainty epoch from the model definition, in which case it would be a requirement of software calculating uncertainty to provide a reference epoch.

New Zealand Geodetic Datum 2000

Brief history

The New Zealand Geodetic Datum 2000 (NZGD2000) was established by Land Information New Zealand (LINZ) in 1998 to replace the existing geodetic datum, NZGD49. NZGD49 was a conventional datum based on terrestrial observations and positioned by astronomic observations. The 1949 datum was a static datum in which the coordinates of approximately 290 first order stations were fixed. Limitations of the observational techniques available for establishing NZGD49 meant that its coordinates were relatively inaccurate compared to modern survey techniques. Also more than 50 years of earth deformation had degraded their accuracy.

NZGD2000 was established as a semi-dynamic datum. This is a datum defined in terms of global reference frame (in this case ITRF96) and a deformation model that relates the NZGD2000 coordinate system to ITRF96. Unlike a static datum no coordinates in New Zealand are fixed. The ITRF geodetic datum realisations are more than accurate and consistent enough for use in New Zealand. However using the deformation model means that the NZGD2000 coordinates objects fixed to the ground in New Zealand do not change significantly. The coordinate system emulates a static datum while being directly based on a dynamic datum.

The original NZGD2000 deformation model was a simple horizontal velocity field defined by a gridded model. This has been updated six times to add deformation due to earthquakes that have occurred since 2000. The first update was in 2013 to account for deformation due to the Christchurch earthquakes of 2010/2011, as well as adding deformation from four other earthquakes that had occurred since 2000.

Most of the updates have been based solely on geophysical models of earthquake faulting developed by GNS Science (a New Zealand geological and geophysical research institute). The two most recent updates have included "refinement grids" which are smoothed models of the difference between the geophysical model and observed deformation (based on GNSS observations). It is anticipated that future updates will include much more input from directly observed deformation derived from radar remote sensing such as INSAR. This is likely to mean much more rapid updates following an update, and more models to better reflect the post-seismic development of the displacement field. Each update of the deformation model is identified by a version number reflecting the approximate publication date of the update in the format YYYYMMDD. The process of updating the deformation model is described in a paper by Winefield, Crook, and Beavan (2010) (https://www.linz.govt.nz/system/files_force/media/file-attachments/winefield-crook-beavan-application-localised-deformation-model-after-earthquake.pdf)

While some users directly access the datum using GNSS observation many datasets are surveyed referencing local survey marks. Recently the survey network coordinates have been maintained by a "national geodetic adjustment" - a least squares calculation of coordinates which uses the deformation model to combine survey observations over the last century, Currently this is run on an as required basis every year or so. The adjustment is constrained by the current ITRF coordinates of approximately 40 continuously operating reference stations (CORS) evaluated at an epoch close to the time of the adjustment.

Patching the NZGD2000 deformation model

The updates to the deformation are applied as patches, multi-resolution gridded models of co-seismic and (in two cases) post-seismic ground deformation. The deformation is applied in one of three ways:

- simple "forward" patches, in which case the deformation added the model and applies for dates after the earthquake
- "reverse" patches, in which case the deformation is added to the station coordinates. A negative deformation is added to the deformation model to apply for dates before the earthquake.

- "hybrid" patch, in which case the deformation is distributed between forward and reverse patches. In this case the reverse patch is used close to the fault where the deformation introduces a lot of distortion, whereas the forward patch is used in the far field where the deformation is very smooth. This means that only datasets close to the faulting need coordinate updates.

Vertical deformation is always applied as a reverse patch so that the height coordinates reflect the current height of marks (ref Stanaway ?).

Publication of the deformation model

Since the first update of the deformation model in 2013 LINZ has published the model as a set of CSV (comma separated value) text files, one for each grid of displacement values, and one defining the time model and other attributes of each component. The current model can be downloaded from https://www.geodesy.linz.govt.nz/download/nzgd2000_deformation_model. This proposal is adapted from that format. However by using binary grid files the format becomes directly and efficiently usable by software.

Each updated has also been supported with an NTv2 grid defining the horizontal reverse patch deformation and a CSV formatted grid file defining the vertical reverse patch. These have allowed users to apply the reverse patch update to data sets.

Strictly speaking every update of the deformation model instantiates a new version of the datum, and a reverse patch creates a new version of the coordinate system (ie the coordinates of "ground fixed" objects are changed).

Up till now however LINZ has not defined new versions of the NZGD2000 datum and related coordinate reference systems (CRS) - as far as GIS software are concerned it is still just one datum and one CRS. This is a pragmatic approach chosen because:

- currently no software other than LINZ tools supports the deformation model
- it takes a long time for a new coordinate system (such as a new version of NZGD2000) to be propagated to GIS software and other tools
- updates have only a small area of the country

This is not ideal, in particular it does not ensure good metadata with a data set about which reverse patch updates have been applied. None the less there have been few or no issues in the GIS community from this potential error source.

However this is changing. It will probably soon be practical to publish a new version of the NZGD2000 CRS for each update to the deformation model and have that change rapidly propagated through the GIS infrastructure. The new CRS (and related projection coordinate systems) as well as the corresponding transformation parameters will be immediately (or at least quickly) available from central repositories. GIS datasets will have a clearly identifiable version allowing users to be confident which patches have been applied.

Greenland (Kristian Evers <kreve@sdfe.dk>)

Greenland is subject to significant elastic deformation due to rapid mass-loss from the ice sheet. The deformation varies a lot from year to year which creates a need for a rather complicated deformation model. At the moment I have solved the problem by creating a set deformation models that cover a specific year each. That is, for every year since 1996 I have a 3D deformation model that only applies for that particular year. The plan is to create a pipeline of deformation operations that applies as many adjustments from the deformation models as needed. E.g. if a coordinate from 2010 is transformed it would have to pass through 14 deformation operations (2010-1996) to account for all the deformation. I think for the final transformations I will narrow the number of gridded models down so that they each cover a three year period. The point here is that it would be nice to be able to support this type of transformation in the new deformation model format. If I read the proposal correctly this would be possible but I may have missed some details. PROJ is actually not able to do this as of now but I will extend the deformation operation to allow this in the future (functionality similar to the +t_epoch/+t_final setup in hgridshift is needed).

Iceland (Kristian Evers <kreve@sdfe.dk>)

Similarly to New Zealand, Iceland exists in very a geodynamically complicated setting. Transformations in Iceland should ideally take care of secular (GIA, divergent plate boundary) and episodic deformation (earthquakes, volcanism) as well as the regular Helmert transformations involving ITRFxxxx. In PROJ this can be handled as a pipeline including the helmert, hgridshift, vgridshift and deformation operations. I gave a talk on the subject last year which covers this use case and some thoughts about a practical implementation [2]. Recently that work described in that presentation has culminated in a pull request for the proj-datumgrid repository [3].

Japan (Tatsuya Yamashita, GSI of Japan <yamashita-t96rq@mlit.go.jp>)

The Geospatial Information Authority of Japan (GSI of Japan) maintains a semi-dynamic datum, Geodetic Datum of Japan 2011 (the Japanese name translates literally as Survey Results 2011). The datum is realised using observations of approximately 1300 CORS stations and was referenced to ITRF2005 at epoch January 1 1997 .

The 2011 Tōhoku earthquake (Mw 9.0) off the Pacific coast of Japan caused extensive deformation to the eastern half of Japan. In response GSI re-realised the datum in the eastern half to a reference epoch of May 24 2011. The transition between the eastern and western halves distributed across a buffer zone that is approximately 200km by 300km.

The ongoing secular deformation is monitored by the CORS network, which is used to generate annual correction grids used to convert from current ITRF coordinates to the datum coordinates. Each grid is developed by interpolating from the coordinate corrections measured at the CORS stations onto a grid of approximately 5km spacing. Each grid applies for a period from the beginning of April to the end of March in the following year.

When earthquakes have occurred since the datum epoch, the co-seismic and immediate post-seismic deformation due to the earthquake has been estimated and added to the datum coordinates. Episodic deformation is modelled on a grid with approximately 1km spacing. This deformation is not included in the annual correction grids, although any residual post-seismic deformation will be.

GSI of Japan provides an online tool that can be used to convert an ITRF coordinate observed at a specific epoch to the corresponding datum coordinate. If the conversion epoch is before an earthquake then the deformation due to the earthquake must be added to the coordinate in addition applying the annual correction for the conversion epoch.

This approach is very similar to the New Zealand Geodetic Datum. The main difference is GSI of Japan can take advantage of the dense CORS network it has established to directly measure the secular deformation, whereas in New Zealand the secular deformation is estimated as a velocity grid based on comparatively sparse CORS data supplemented with about 10 years of GNSS campaign data.

North America

(TBC - capturing here notes based on comments from Kevin Kelly, Jeff Freymueller, and Chris Pearson)

NGS has modelled a post-seismic deformation field including a time-function for the 2002 Mw 7.9 Denali earthquake.
See:https://www.ngs.noaa.gov/TOOLS/Htdp/HTDP-user-guide.pdf and Snay RA, Freymueller JT, Pearson C (2013) Crustal motion models developed for version 3.2 of the Horizontal Time-Dependent Positioning utility, J. Appl. Geod., 7:173-190, doi: 10.1515/jag2013-0005

See also Snay, R. A., J. T. Freymueller,M. R. Craymer, C. F. Pearson, andJ. Saleh (2016), Modeling 3-D crustal velocities in the United States and Canada,J. Geophys. Res. Solid Earth,121,5365–5388, doi:10.1002/2016JB012884.
(https://agupubs.onlinelibrary.wiley.com/doi/epdf/10.1002/2016JB012884)