Uncertainty Metadata Naming Conventions (UNC) Specification

version v0.1

CoMet Toolkit Team

October 27, 2026

Contents

Uncertainty Metadata Naming Conventions (UNC) 1

- Authors 1
  - Lead Authors 1
- Introduction 1
  - Goals 1
  - Philosophy 1
  - Terminology 2
  - Format for Examples 2
- Measurement Dataset Structures 2
  - Variables 2
  - Dimensions 3
  - Data Types 3
  - Attributes 3
- Uncertainty Attributes 3
  - Assigning Uncertainty Components 3
  - Units 4
  - Uncertainty PDF Shape 4
  - Error-Correlation Structure 5
    - Parameterisating Error-Correlation Matrices 5
    - Storing Error-Correlation Parameterisation 5
- Appendix A: Error-Correlation Parameterisations 6
  - Parameterisations based on Matrix Structure 6
  - Parameterisations based on Data 7

Uncertainty Metadata Naming Conventions (UNC)

Authors

Lead Authors

- Sam Hunt, NPL
- Pieter De Vis, NPL

Introduction

Goals

Measurement datasets are becoming larger, more complex, and are increasingly used to support critical applications such as manufacturing, health, and environmental monitoring. Reliable interpretation of these measurements requires accompanying uncertainty and error-covariance information - however, this is often overlooked. Where available, such information lacks standardisation and could, in principle, be highly complex and large.

The goal of this Uncertainty metadata Naming Conventions (UNC) specification is to provide a standardised metadata format for storing the accompanying uncertainty/error-covariance information with measurement datasets. This format is intended to support fully capturing the content of the error-covariance matrices associated with measurement data in a compact structure, by parameterising error-covariance with a simple set of metadata.

Philosophy

This specification is intended to contribute to and build upon an existing ecosystem of standards and best practices. In particular the following are adhered to, to the extent possible:

- The understanding of uncertainty concepts defined in the JGCM GUM (Guide to the expression of uncertainty in measurement) suite of documents.
- The definition of uncertainty-related terminology defined in the JGCM VIM (International Vocabulary for Metrology).
- The NetCDF data model for creating self-describing, array-oriented scientific datasets.
- The Climate and Forecast (CF) conventions on metadata for weather and climate data.

The work builds on previous work on the standardisation uncertainty information for climate data developed within the H2020 FIDUCEO project.

Within this context, this specification also attempts to adhere to the following principles:

- **Scalability of Complexity**

  The complexity of the metadata should align with the use case:

  - *Simple use cases* should be achievable with a *simple implementation*.
  - *Complex use cases* should be *possible without unnecessary restrictions*.
- **Minimisation of Redundancy**

  The metadata specification should *avoid duplication* of information to prevent potential inconsistencies.
- **Human and Machine Readability**

  Metadata must be:

  - *Comprehensible* for humans.
  - *Parsable* by machines.

Terminology

For terminology related to measurements and associated uncertainties, definitions within the VIM (International Vocabulary for Metrology) are adopted to the extent possible. This includes the following important terms:

- Error
- Uncertainty
- Coverage factor

Definitions of the following terms are adopted from work undertaken in the H2020 FIDUCEO:

- Error-correlation
- Error-covariance

Note

Are there any other widely adopted definitions we can use for these terms?

Format for Examples

The plain text (ASCII) format used to describe the contents of a NetCDF dataset is called CDL (NetCDF Common Data Language). Snippets of CDL are used to present examples in this specification. A minimal example of a measurement dataset in CDL is given below. Here the dataset contains a single `temperature` variable with accompanying metadata (`units` and `description`), which is defined along `time`, `lat` and `lon` dimensions (each of which is length 2). Dimension sizes and data are omitted in specification examples for brevity.

```
netcdf short_example {
  dimensions:
    time = 2 ;
    lat = 2 ;
    lon = 2 ;

  variables:
    float temperature(time, lat, lon) ;
      temperature:units = "K" ;
      temperature:description = "measured temperature" ;

  data:
    temperature =
      290.1, 291.2,
      292.3, 293.4,
      294.5, 295.6,
      296.7, 297.8 ;
}
```

Measurement Dataset Structures

As mentioned above, the NetCDF data model for creating self-describing, array-oriented scientific datasets is adopted. The components of NetCDF datasets are described in Section 2 of the NUG (NetCDF Users Guide). In this section, we introduce the core components of this data model relevant to this specification.

Variables

Datasets are composed of variables, which are multidimensional data arrays.

This specification defines the following categories of variables:

- **Observation Variables**
  Observation variables represent a multidimensional array of measurements.

- **Uncertainty Variables**

  Uncertainty variables represent a component of uncertainty associated with an *observation variable*. An *observation variable* may have multiple *uncertainty variables* associated with them.

  Uncertainty variables must have the same dimensions as the *observation variable* they are associated with.

> **Note**
>
> Should we allow uncertainty variables to be smaller that observation variables? i.e. that have a subset of the dimensions to save space where there are repeated values? (in practice, compression would reduce this as well…)

A dataset may also contain variables that are neither *observation variables* or *uncertainty variables*.

Dimensions

A variable may have any number of named dimensions, including zero – e.g., `"x"`, `"y"`, `"time"`. Dimensions may be of any size, including 1.

Data Types

> **Note**
>
> Can we permit different types?

Observation variables and *uncertainty variables* must be `floats`.

Note: these variables may be encoded as e.g. integers for efficient storage on disc.

Attributes

Dataset attributes provide metadata about the dataset, its variables, and dimensions. Global attributes describe the entire dataset (e.g., title, institution, history). Variable attributes define specific properties of the variable (e.g., units, valid ranges). These attributes ensure data is interpretable, support automated processing, and facilitate sharing by following standardised conventions.

This specification defines a set of variable attributes to:

- link *observation variables* with their associated *uncertainty variables*
- define the error-correlation properties of a given *uncertainty variables* in a compact way.

A dataset may also contain other unrelated attributes.

Uncertainty Attributes

Assigning Uncertainty Components

Uncertainty variables are associated with their *observation variable* through the *observation variable*'s `"unc_comps"` attribute. This attribute contains a list of the names of all of the *uncertainty variables* associated with an *observation variable*.

The following example of a dataset, in CDL syntax, shows a `temperature` variable defined along 3 dimensions - `time`, `lat`, and `lon`. `temperature` has two uncertainty components associated with it - `u_calibration` and `u_noise`.

```
variables:
  float temperature(time, lat, lon) ;
```

```
    temperature:unc_comps=["u_calibration", "u_noise"];
  float u_calibration(time, lat, lon);
  float u_noise(time, lat, lon);
```

Units

The physical units associated with *observation variables* and *uncertainty variables* should be defined by the `"units"` variable attribute as a string.

Observation variables are assumed dimensionless if the variable attribute `"units"` is not defined.

uncertainty variables must have the same `"units"` as the *observation variables* they are associated with. If `"units"` is not defined, the *uncertainty variable* is assumed fractional.

The following example dataset again shows a `temperature` variable associated with two uncertainty components - `u_calibration` and `u_noise`. Here, `u_calibration` is defined with units `K`, matching `temperature`. `u_noise` has no defined units and so assumed is a fractional uncertainty.

```
variables:
  float temperature(time, lat, lon);
    temperature:unc_comps=["u_calibration", "u_noise"];
    temperature:units="K"
  float u_calibration(time, lat, lon);
    u_calibration:units="K"
  float u_noise(time, lat, lon);
```

Uncertainty PDF Shape

The probability density function (PDF) shape associated with the uncertainty estimate values in an *uncertainty variable* is defined with the variable attribute `"pdf_shape"`.

`"pdf_shape"` can have one of the following values:

- `"gaussian"` - for uncertainties represented by a Gaussian PDF
- `"rectangular"` - for uncertainties represented by a uniform PDF
- ...

Note

What PDF shapes should we allow? Is there a list somewhere else we can refer to?

If `"pdf_shape"` is not defined for an *uncertainty variable* it is assumed to be `"gaussian"`.

The following example dataset again shows a `temperature` variable associated with two uncertainty components - `u_calibration` and `u_noise`. Here, `u_calibration` is defined to be represented by a rectangular PDF. `u_noise` has no defined `"pdf_shape"` and so is assumed Gaussian.

```
variables:
  float temperature(time, lat, lon);
    temperature:unc_comps=["u_calibration", "u_noise"];
    temperature:units="K"
  float u_calibration(time, lat, lon);
    u_calibration:units="K"
    u_calibration:pdf_shape="rectangular"
  float u_noise(time, lat, lon);
```

Error-Correlation Structure

To provide the complete uncertainty information associated with an *observation variable*, the cross-element error-covariance matrix is required. In practice, error-covariance matrices are often determined from a combination of the per element uncertainties (i.e., the *uncertainty variable* described above) and the cross-element error-correlation matrix. This section therefore defines a standardised way to store error-correlation matrices, to enable this complete description of dataset error-covariance.

Parameterisating Error-Correlation Matrices

For *observation variables* with *N* elements, the associated error-correlation matrix per *uncertainty variable* has the square of *N* elements. Where *observation variables* are large, it quickly becomes impractical to store this data. However, in many cases the associated error-correlation matrix can in fact be simply parameterised in a compact form, e.g., by defining it as having a specific structure (i.e., identity, full, banded).

Such a parameterisation is here defined by 3 values, as follows:

- `form` - the parameterisation name, which defines the functional form of the parameterisation (e.g., “random” for identity matrix structure).
- `params` - a list of parameters associated with the parameterisation (e.g., the bandwidth for a banded matrix).
- `units` - the physical units associated with each parameter in `params` list.

Appendix A defines a set of standard error-correlation parameterisations using this structure. For cases not covered by this standard set, users may make their own definitions and propose them for inclusion. The simplest and most commonly used parameterisations from this standard set are:

Simple Error-correlation parameterisations

Form	Parameters	Matrix Structure	Description
`random`	None	Identity Matrix	No error-correlation between elements in observation variable.
`systematic`	None	Full matrix of 1’s	Full error-correlation between elements in observation variable.
`err_corr_matrix`	[`err_corr_matrix_var`]	Explicitly defined matrix	User defined error-correlation matrix, stored in data variable with name `err_corr_matrix_var`.

To allow maximum flexibility, different parameterisations can be defined along each *observation variable* dimension, `dim_x`, or sets of dimensions, `[dim_x, dim_y, ...]`. For example, an error could be fully correlated in longitude and latitude at each time step, but uncorrelated between time steps.

Note

Pieter can we add the maths of how these sub-error-correlation matrices combine?

Storing Error-Correlation Parameterisation

This specification defines a set of *uncertainty variable* variable attributes to store these error-correlation parameterisation values.

The following *uncertainty variable* variable attributes are defined to store the parameterisation information, per dimension or set of dimensions (each labelled by `i`, which runs from 1 to the required number of dimensions / sets of dimensions):

Error-correlation *uncertainty variable* variable attributes

Attribute name	Type	Description	Example
`err_corr_dimi_name`	`str`	Dimension name	`err_corr_dim1_name="time"`
`err_corr_dimi_form`	`str`	Parameterisation `form`	`err_corr_dim1_form="random"`
`err_corr_dimi_params`	`list[str | float | int]`	Parameterisation `params` (can omit if none required)	`err_corr_dim1_params=[1,2,3]`
`err_corr_dimi_units`	`list[str]`	Parameterisation `units` (can omit if none required)	`err_corr_dim1_params=["second", "K"]`

The following example dataset again shows a `"temperature"` variable associated with two uncertainty components - `"u_calibration"` and `"u_noise"`.

Here, `"u_calibration"` is defined to have a systematic error-correlation in the `lat` and `lon` dimensions. In the `time` dimension, an explicit error-correlation matrix is defined with the variable `err_corr_calibration_time`.

`"u_noise"` has is defined has having uncorrelated errors in all dimensions.

```
variables:
  float temperature(time, lat, lon);
    temperature:unc_comps=["u_calibration", "u_noise"];
    temperature:units="K"
  float u_calibration(time, lat, lon);
    u_calibration:units="K";
    u_calibration:pdf_shape="rectangular";
    u_calibration:err_corr_dim1_name=["lat", "lon"];
    u_calibration:err_corr_dim1_form="systematic";
    u_calibration:err_corr_dim2_name="time";
    u_calibration:err_corr_dim2_form="err_corr_matrix";
    u_calibration:err_corr_dim2_params=["err_corr_calibration_time"];
  float u_noise(time, lat, lon);
    u_calibration:err_corr_dim1_name=["time", "lat", "lon"];
    u_calibration:err_corr_dim1_form="random";
  float err_corr_calibration_time(time, time);
```

Appendix A: Error-Correlation Parameterisations

This Appendix defines a set of standard parameterisations for error-correlation matrices. As described above, these parameterisations are defined with 3 values, as follows:

- `form` - the parameterisation name, which defines the functional form of the parameterisation (e.g., “random” for identity matrix)
- `params` - a list of parameters associated with the parameterisation e.g., the bandwidth for a banded matrix
- `units` - the physical units associated with each parameter in `params` list

Parameterisations based on Matrix Structure

The following defined error-correlation matrix parameterisations are based on standardised matrix structures (e.g., identity, full). The following table defines the `form` and `params` necessary to define matrices of a given structure.

Note

I know Zhav was unsuccessful in finding a standard set of matrix structures we can adhere to… but I’m interested in any similar suggestions to simplify this definition…

Form	Parameters	Matrix Structure	Description
`random`	None required	Identity matrix	No error-correlation between elements in the *observation variable*.
`systematic`	None required	Full matrix of 1's	Full error-correlation between elements in *observation variable*.

Parameterisations based on Data

The following defined error-correlation matrix parameterisations are based on providing data that defines the contents of the matrix. The following table defines the `form` and `params` of these parameterisations.

Note

Anything urgent to add? Sparse matrix definitions?

Form	Parameters	Matrix Structure	Description
`err_corr_matrix`	`err_corr_matrix_var` - name of the dataset variable that defines the error correlation matrix.	Explicitly defined matrix	Full error-correlation between elements in *observation variable*.